NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
GODDARD SPACE FLIGHT CENTER

BAT Flight Software Requirements Document

Version 1.5b
410.4-RQMT-0009

Apr 02, 2001

BAT Flight Software Requirements Document

Version 1.5b, Apr 02, 2001

SIGNATURE PAGE

Michael D. Blau — 4/11/01
Mike Blau — Date
NASA GSFC Code 582 / BAT C&DH Software Lead

Ed Fenimore — Date 4/10/01
Ed Fenimore
Los Alamos National Laboratory / BAT Science Code Lead

Elaine Shell — 4/17/01
Elaine Shell — Date
NASA GSFC Code 582 / Flight Software Branch Head

Beverly Settles — 4/17/01
Beverly Settles — Date
NASA GSFC Code 561 / BAT IP Hardware Lead

— 4-18-01
Frank Kirchman — Date
NASA GSFC Code 740 / BAT Systems Lead

— 17 Apr 01
Scott Barthelmy — Date
NASA GSFC Code 661 / BAT Science Systems Lead

Lawrence M. Hilliard — 4/17/2001
Larry Hilliard — Date
NASA GSFC Code 740 / BAT Instrument Manager

John C Ong — 4/17/01
John Ong — Date
NASA GSFC Code 582 / Swift Software Systems Manager

TABLE OF CONTENTS

1 INTRODUCTION

1.1 SCOPE AND PURPOSE

This document forms the basis for the design of the Burst Alert Telescope (BAT) flight software for the Swift mission. The purpose of this document is to specify all of the stated requirements as well as the functional and performance requirements that have been derived from previous mission design activities.

1.2 DOCUMENT OVERVIEW

This document is divided into three main sections. The first section is introductory in nature and is intended to give the reader background information on the system specification and architecture. This provides a context in which the reader can apply the binding requirements specified in the rest of the document. The remainder of the document specifies the requirements imposed upon the design of the flight software.

1.3 TRACKING REQUIREMENTS

Each functional requirement (in Sections 3 and 4 of this document) has been assigned a unique number. These numbers will be used to build traceability and testability matrices during the test and validation phases of the flight software development life cycle to verify that each requirement has been satisfied and performs according to the specification. Successful completion of the Acceptance Tests will qualify the software as *ready* for Instrument environmental testing.

The requirements listed in this document have been grouped according to functional areas. The entire group of requirements forms an outline with individual subsections being assigned a number unique to their place in the outline. As a result, each requirement has its own number identifying its place in the outline. These are the numbers that should be used in the aforementioned matrices.

1.4 APPLICABLE DOCUMENTS

- Swift Mission Requirements Document
- Swift Interface Requirements Document
- BAT Science Requirements Document
- Swift BAT Instrument Flight Software Product Plan
- Swift 1553 Bus Protocol Interface Control Document
- BAT Science Software to C&DH Software ICD
- Swift FoM Software Requirements Document
- Swift BAT IPE to BAT Power Box Interface Control Document
- Swift-BAT BCDH Software ICD
- BAT DMC Specification

- CCSDS 701.0-B-2 Advanced Orbiting Systems, Networks and Data Links: Architectural Specification. Recommendation for Space Data Systems Standards. Blue Book. Issue 2. Washington, D.C.: CCSDS, November 1992.

- CCSDS 201.0-B-2 Telecommand, Part 1: Channel Service. Blue Book. Issue 2. November 1995.

- CCSDS 202.0-B-2 Telecommand, Part 2: Data Routing Service. Blue Book. Issue 2. November 1992.

- CCSDS 202.1-B-1 Telecommand, Part 2.1: Command Operation Procedures. Blue Book. Issue 1. October 1991.

- CCSDS 203.0-B-1 Telecommand, Part 3: Data Management Service. Blue Book. Issue 1. January 1987

1.5 ACRONYMS

ADU	Analog-to-Digital Unit
ATS	Absolute Time Sequence
BAT	Burst Alert Telescope
BATSE	Burst and Transient Source Experiment
BCDH	Block Command and Data Handler
BSC	BAT Science Code
C&DH	Command & Data Handling
CCSDS	Consultative Committee for Space Data Systems
DAP	Detector Array Plate
DMA	Direct Memory Access
DRAM	Dynamic RAM
DSP	Digital signal processor
ECC	Error Correction Code
EDAC	Error Detection and Correction
EEPROM	Electronically Erasable Read Only Memory
FIFO	First-In First-Out
FOT	Flight Operations Team
FoM	Figure of merit
GCN	GRB Coordinates Network
GRB	Gamma Ray Burst
GSFC	Goddard Space Flight Center
ITOS	Integration, Test, and Operations System
I/F	Interface
I&T	Integration and Test
ICD	Interface Control Document
IPE	Image Processing Electronics
Kb	Kilobyte
LIO	Local Input/Output
LVPC	Low Voltage Power Converter
Mb	Megabit
MB	Megabyte
MET	Mission Elapsed Timer
MHz	Megahertz

NASA	National Aeronautics and Space Administration
PCI	Peripheral Component Interconnect (Bus)
PPS	Pulse Per Second
QHSS	Quad High Speed Serial Interface
RAM	Random Access Memory
ROM	Read Only Memory
RT	Remote Terminal
RTS	Relative Time Sequence
S/C	Spacecraft
SCS	Stored Command Sequence = RTS or ATS
SEU	Single Event Upset
SMEX	Small Explorer
SRAM	Static RAM
SRR	System Requirements Review
SUROM	Start-Up ROM
TBD	To Be Determined
TBR	To Be Required
TBS	To Be Specified
TDRSS	Tracking and Data Relay Satellite System

2 FLIGHT SOFTWARE OVERVIEW

The design and development of the flight software is governed not only by the requirements specified in this document, but also by the chosen system architecture and the intended environment. This section introduces the flight software within the context of these topics.

The BAT top level requirements are defined in the BAT System Requirements Document and the BAT System Requirements Review (SRR) data package. The upper level requirements are restated below for convenience.

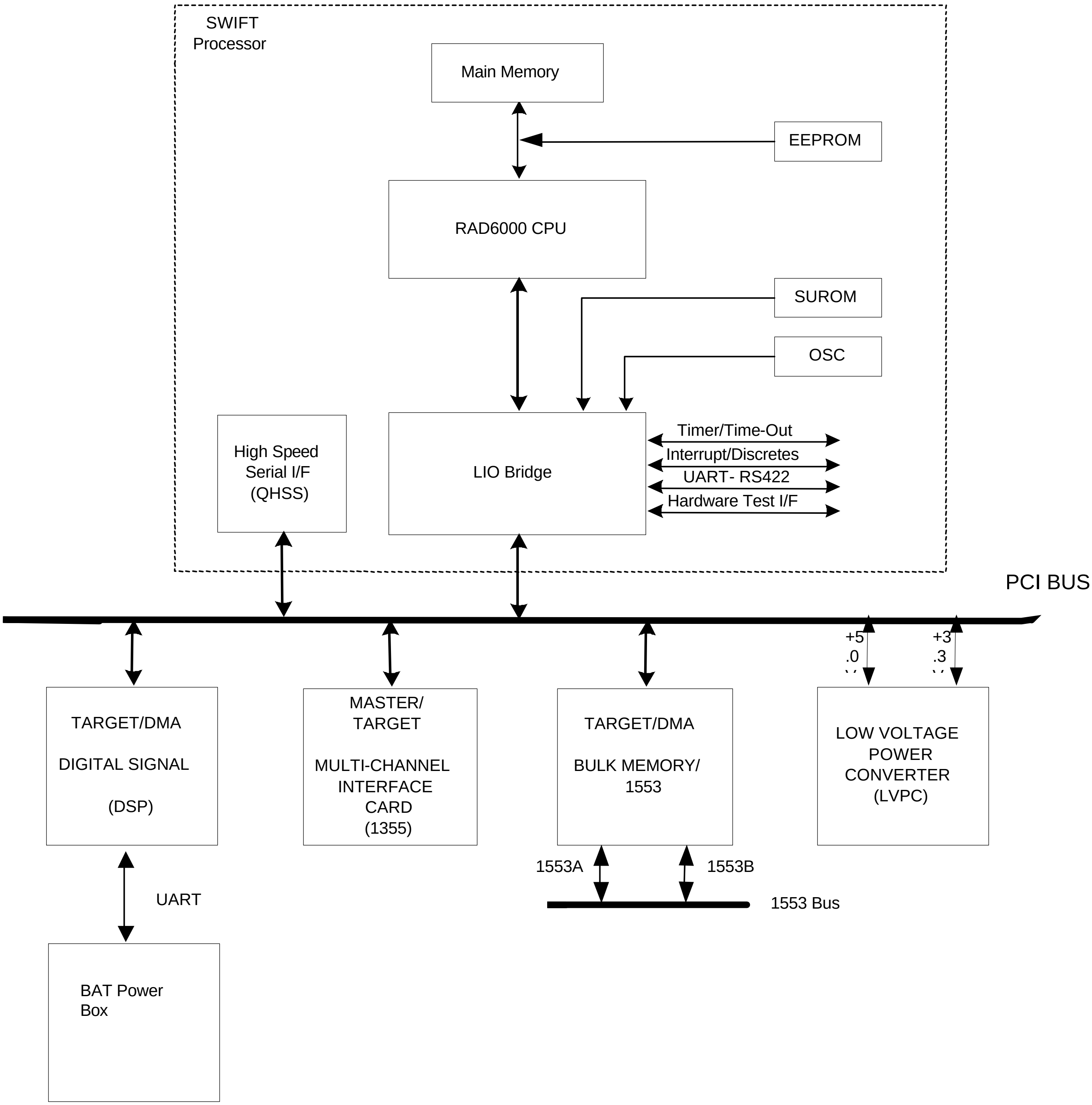
BAT Requirements

1. Observe a large number (~300) of bursts per year
2. Detect bursts 5x fainter than BATSE
3. Have a low energy threshold to discover soft GRBs
4. Have a bright GRB capability
5. Determine GRB positions to within 4 arcmin
6. Compute GRB positions in less than 6 seconds
7. Measure a gamma-ray light curve for each GRB (including both precursor & post-burst data
8. Perform an all-sky hard x-ray survey
9. Shall perform as a hard x-ray transient monitor

2.1 FLIGHT SOFTWARE CONTEXT

The Swift BAT Flight Software will run on two processors: A RAD6000 microprocessor at 25MHz, and a 21020 digital signal processor (DSP) at 18MHz. Figure 1.0 below shows a representation of the Swift architecture. Flight Software applications for the RAD6000 will be developed using the "C++", and "C" Programming languages. The Operating System is the Wind Rivers VxWorks/Tornado version 1.0.1.

Flight software applications for the 21020 DSP will use the Analog Devices ADSP-21000 Family software development tools, with code written in C and assembly language.



The Swift BAT information processing electronics (IPE) will be a RAD6000 processor card with four custom electronics cards:

- 1553/DRAM/EEPROM card
- 21020 DSP card
- Multi-channel I/F card.
- LVPC card

The custom cards are designed by GSFC Code 561

2.2 MAJOR SOFTWARE COMPONENTS

Three major software components will execute within the BAT IPE: BAT Science code, Bat C&DH code, and the Figure of Merit code. These interfaces will be documented in BAT Science Software to C&DH Software ICD.

The BAT IPE software will consist of the following software elements:

- Science processing
    - Survey Mode
    - Burst Mode
    - Alerts
    - \- Trigger
    - \- Burst
    - Data Products
- Figure of merit processing*
- C&DH processing
    - command processing
    - telemetry processing
    - bulk memory data storage
    - housekeeping status reporting
    - inter-task communications
    - external 1553 bus remote terminal
    - stored command processing
    - telemetry limit checking
    - system activity scheduling
    - software task management
    - onboard software maintenance
    - onboard file system management
    - power management
    - memory checksumming
    - bulk memory scrubbing
    - onboard self test diagnostics
    - science data processing

*Requirements for the figure of merit software are documented in a separate requirements document.

2.3 SYSTEM DESIGN CRITERIA

This section specifies general design goals for the software. These goals are not to be considered as testable requirements by themselves.

-1 The design and implementation of the flight software shall be based on the Triana software. It shall be a goal to reuse as much of the Triana software as practical. Lessons learned corrections and improvements should be incorporated where appropriate.

-2 It shall be a design goal for the flight software to maintain as much compatibility as possible with telemetry and command formats used in prior SMEX missions.

-3 As a design goal, the instrument shall be capable of autonomously functioning for 1 week.

-4 As a goal, the flight software shall use system tables wherever feasible to define system configuration and operations data.

3 BAT SCIENCE CODE REQUIREMENTS

3.1 DESCRIPTION OF MINIMUM CORE REQUIREMENTS AND VERIFICATION

In the following sub-sections for section 3, the requirements are classified as (a) Minimum Core Requirement, and (b) Strongly Desired Requirements. For those items which are classified as Minimum Core Requirement, they are identified with the indicator "[MinCoreReq]".

A wide array of techniques and procedures will be used to validate and verify the requirements listed in section 3. These include (a) direct testing of a specific functionality, (b) indirect testing of a specific functionality via a general procedure which uses specific functionality indicated as part of a series of functions and capabilities (i.e. it the total procedure works, then the individual parts work), and (c) validation by analysis. As part of the Science Code flight s/w, many data simulation tools will be developed. These tools will allow the s/w tester to generate controlled and repeatable data streams from the Blocks that will allow testing of the flight s/w for the MIC, DRAM, CPU, and DSP parts of the Image Processor. The later method, "c", will be used sparingly, but is necessary for some functionalities which are just too complicated to handle with the direct approaches.

3.2 GENERAL INFORMATION AND FUNCTIONAL DESCRIPTION

The BAT Science Code (BSC) consists of at least 8 functional categories [all MinCoreReq]. These are listed and described here as an overview to the "science code" portion of the flight s/w. They should not be considered to be "requirements" in the technical sense, and hence do not require validation and verification procedures -- they are a descriptive overview. The specific requirements are listed in later sub-sections of section 3, and those specific requirements will be subjected to verification and validation.

1) The *Data Ingest Category* continually reads the most recent data from the Detector Array and converts it into these internal data products:
- Converts the raw pulse height of the photon event data into true energy scale.
- Detector Time Histories: 9 detector areas, in 4 E-bins, at 4 ms resolution.
- Mask-Tagged Time Histories: 4 E-bins, 64 ms resolution, per source.
- Pulsar folds: 80 E-bins, 32 phase bins, per source.
- Survey Detector Map: 80 E-bins, ~5-minute accumulations, 32k detectors.
- Background Detector Map: 4 E-bins, quasi-log (8s - full_obs. accumulation, 32k detectors.
- Calibration accumulations (electronic pulser and tagged-source events).
- Housekeeping data (e.g. voltages, currents, temps, rates, status bits, s/w-based HK items, …).

2) The *Trigger Category* searches these internal data products for count rate excesses indicative of Gamma Ray Bursts (GRBs). There are two basic types of triggers: (a) rate triggers and (b) image triggers. This search spans a large phase space with dimensions of:
- energy range (the 4 standard energy intervals)
- order of fit to the background (0^{th}, 1^{st} or 2^{nd} as a function of time)
- duration and offset of background interval(s)
- duration of trigger interval (e.g. 4 msec to 32 sec)
- detector region (e.g. quadrants, halves, and the whole array)
- exclusion of known sources (e.g. Cyg X-1)
- rate trigger and imaging trigger

The Rate Trigger will analyze the above trigger criteria. When it finds data that exceeds one or more of the trigger criteria, it will select an energy range and source and background time intervals based on these criteria to optimize the detection of a GRB by imaging, and will inform the Burst Response task of this selection. The burst trigger criteria will be evaluated for all the rate data except during the burst response (~5 minutes) and during slews (~20-100 seconds). Background fits will exclude data taken with a different attitude or instrument configuration; thus the instrument will not trigger for a criterion-dependent time following slews, SAA passages, Power-Up, and any other time via ground command. The Imaging Trigger searches sky images produced by the DSP from detector maps for unknown sources and interesting fluctuations in known sources. Based on the duration of the increase, the trigger will initiate either a burst or transient response.

3) The *Burst Response Category* will respond to the Trigger Category's wake-up call, and analyze the suggested time & energy intervals and the detector array geometry. It will image that data to produce either a GRB location or a non-GRB conclusion (e.g. "that's not a GRB, that's an electron shower", because it did not produce a point-source in the sky image). After detecting a real GRB, it generates the data packets for transmission to the ground via various routes (TDRSS and SSR). After a non-GRB conclusion, the Burst Response exits and the Trigger resumes scanning all of its trigger criteria looking for a new trigger.

4) The *Survey Products Category* compresses the internal data products into the proper form (i.e. energy spectrum histograms for each of the 32K detectors every ~5 minutes) and sends them to the ground. [Most of the BAT SOH HK is handled by the BAT C&DH code.]

5) The *Calibration Category* uses the internal data products to continuously recalibrate the detector plane. That is, except during a burst response, a portion of the DAP Electronics will be dedicated to the calibration of the XA1 output. For a detailed description of the procedures, types of calibration scenarios and analysis of the calibration data, see the “Operational Modes & Sequences for BAT” and the “Swift BAT Flight Hardware and Software Calibration Functional Requirements”, March 26, 2000 by Scott Barthelmy.

6) The *SOH Category* (a) collects the housekeeping information, (b) passes it on to the BAT C&DH code for (ba) telemetry stream (both the full amount for the stored telemetry and a subset for the real-time telemetry), (bb) checks the values against the acceptable range limits, and (bc) sends the appropriate alarm messages (both the TDRSS alarms and the lower-level "markers" in the regular telemetry stream) for those items outside the acceptable range.

7) The *DSP Category* is the processing of the detector maps (a) from the Burst Response Category for the imaging of the bursts and (b) from the imaging of the survey maps for the Trigger Category looking for the longer timescale hard X-ray transients. This imaging process involves both the FFT method and the Back-projection method. The scanning of the images to identify point sources is done by the Burst Response Category.

8) The *Diagnostic Category* is a set of special modes, functions, and features in the flight software, plus a set of special standalone functions and programs (not part of the flight software) which allow for special data-taking modes to be accomplished.

3.3 GENERAL BAT SCIENCE CODE REQUIREMENTS

-0 These are all MinCoreReq.

-1 The BSC shall be completely re-programmable from the ground.
This includes both the RAD6000 and the DSP, and the RAM & the non-protected portion of the EEPROM on both platforms.

-2 The system, the architecture, and the procedures shall be compatible with remote testing and operations from Los Alamos. This includes the IPE ETU and flight hardware at GSFC.

-3 The BSC shall receive from the engineering code ephemeris knowledge (Long,Lat to an accuracy of 50 km every 0.2 sec).

-4 The BSC shall receive from the engineering code attitude knowledge (all 3 axes orientation to an accuracy of 0.5-1 arcmin every 0.2 sec).

-5 The BSC shall receive from the engineering code knowledge of S/C slew activity (e.g. the message from the s/c ACS telling when the slew has settled to less than 30(?) arcsec on the desired RA,Dec location).

-6 The BSC shall maintain an on-board table of known gamma-ray sources and pulsar information.

-6.1 This known-sources table shall, at a minimum, include:

Description of source including a specification of the class of source (e.g. X-ray nova, SGR, GRB, LMXRB, HMXRB)
Location (J2000 RA,Dec)
Expected Spectrum (1or2 P.L. indices, SCS, …)
Expected Variability (Intensity and other methods)
For each pulsar, a priority and frequency ephemeris (double-precision frequency, frequency derivative, and T0), and a dis/enable flag controlling if the epoch-folding is to be done on this object.
Instructions for the reporting of the source found (the same instructions as for the mask-tagged rates).
A provision for the sun's and moon's position to be stored and used to exclude slew messages to the FOM for sources that are too close to the sun or moon.

-6.2 The BSC shall have the capability to change the contents of the on-board table either by on-board evaluation or by updates from the ground. [A good starting point for generating this table of known sources is from Macom and Gehrels (ApJSupp, v120, pp335, 1999).]
This table will be sent to the ground (SRR) every time it is changed.
This table will also be sent to the ground (SRR) every day.

-7 The BSC shall support diagnostic modes for ground testing, on orbit checkout and troubleshooting. See the details in section 3.10.

-8 The BSC shall be implemented within the architectural and protocol guidelines and rules laid out in the Engineering Software Code. See the ICD for the details.

3.4 DATA INGEST REQUIREMENTS

-1 The BSC shall be capable of sustaining a continuous ingest of 45K events/second (256KB/sec) plus 3.6K packets/sec (16 KB/sec) of non-event data (e.g. HK, command verifies, etc.). These events consist of approximately 34K events/second (136KB/sec) of background data plus approximately 11K events/second (120KB/sec) of calibration data.

-2 The BSC shall be capable of sustaining an increase to 240K events/second for at least a 5 second period (for reference, the 240 K and 5 numbers are derived from the most fluent burst compressed into a 5-sec interval at the BBOY flux). There will be a graceful handling of higher data rates (e.g. as will be encountered inside the SAA). Examples of graceful methods are: (a) disabling the event generation at the DM-level and/or at the Block-level, or (b) not performing the full processing on the events (e.g. no histogram accumulation), or (c) processing only a subset of events (e.g. processing only 1 of N events).

-3 There shall be no loss of the housekeeping data that is also present in the data streams from the Blocks (in particular the event counter packets).

-4 The BSC shall calculate proximity to the SAA no greater than every 2 sec (this corresponds to 0.13 degrees in Longitude or 14 km in distance).

5 The BSC shall resume normal operations of the array and processing of data from the array when the event counts and the ephemeris data indicate that the s/c has exited the SAA (i.e. this is a check of the 5-Hz ACS Message Long,Lat against the SAA polygon maintained in the BAT flight s/w).

-6 The Data Ingest functionality shall distinguish between (a) gamma-ray events, (b) tagged-source events, and (c) calibration events and separately accumulate the summaries of these needed by the other categories of processing (*e.g.*, event rate summaries for the Trigger task, histograms of tagged events and mean & sigma of calibration events for the Calibration task).

-7 There shall be a method to throttle the Data Ingest function to limit throughput such that the other tasks/activities within the RAD6000 have sufficient cpu resources (time) to complete their requirements. This throttling shall not be activated during bright bursts (e.g. BBOY), but will activate during long periods (>20 sec) of high count rate (e.g. Solar Flares, electron precipitation events, etc.). This throttling method shall be adjustable via ground command. The throttling method is probably based on a percentage-of-the-ringbuffer-used metric. A typical threshold for activating the throttling will be around 20% (e.g. 20% is 60sec at nominal background event rate which would take 5-10sec of a fully utilized cpu (assuming the nominal cpu utilization is about 10-20%)). Triggering may be disabled when throttling occurs.

-8 From the gamma-ray event data, Data Ingest shall build and maintain, at a minimum, the following internal data products:

True Energy Scale: it converts the raw pulse heights of the photon events from ADUs to keV using the offset and gain values for each detector as determined by the Calibration Task. [MinCoreReq]

Detector Time Histories: ~36 (4 halves+4quadrants+whole_array=9 detector areas in 4 E-bins with 4 ms resolution). Used for Rate Trigger searches and (rebinned at lower resolution due to bandwidth limits) sent to the ground for further scientific and engineering analysis. [MinCoreReq]

Mask-Tagged Time Histories: 4 E-bins, 64 ms resolution, per source.
Used for rejection of false triggers from variable sources, and (rebinned) send to the ground for scientific analysis. There will be a table specifying which locations

are to have the mask-tagging calculation executed. This table will be sent to the ground (SRR) every time it is changed and periodically once every day. [The functionality is MinCoreReq, and the MinCoreReq for the quantity of histories is 1.]

Pulsar folds: 80 E-bins, 32 phase bins, per source for up to 6 (commandable, subject to CPU availability) sources. These pulsars are selected to be the pulsars in the FOV with the highest priority in the known-sources table. [Not MinCoreReq]

Survey Detector Map: 80 E-bins, 5-minute accumulation, 32K dets. The edge locations, widths, and number of bits allocated for each of the 80 E-bins is adjustable. The number 80 may be changed, but a software upload and reboot will be necessary. These E-bin will be specified in a table. This table will be sent to the ground (SRR) every time it is changed and periodically once every day. [Both the functionality and the numbers are MinCoreReq.]

Background Detector Map: 4 E-bins, quasi-log (8s - full_obs. accumulation, 32K detectors. . [Both the functionality and the numbers are MinCoreReq.]

Calibration accumulations (e-pulser events and tagged-source events). [The Offset and Gain cycles are MinCoreReq along with the Tagged Source capability is MinCoreReq. The other e-pulser cycles are not MinCoreReq -- see the details in section 3.8.]

BCDH Housekeeping data (e.g. voltages, currents, temperatures, rates, logical status bits, s/w-based HK items, …). [MinCoreReq]

-9 The above internal data products will be made available to the remaining functions within the BSC.

-10 The Data Ingest functionality shall create data products for the event and diagnostic housekeeping streaming functions.

-11 The Data Ingest will notify the other tasks of the completion of other data products as appropriate (*e.g.* notify the Image Trigger Task upon completion of the 1-min, 5-min, and full-observation detector arrays; the Calibration activity may be synchronized to survey accumulations, and will be synchronized by the receipt of the DM mode change packet form the BCDH indicating the end of calibration; the Survey activity will compress and transmit each Survey Detector Map after the accumulation is complete.

-12 The Data Ingest shall, upon any instance of a DM being turned off by the hardware-based latch-up detection circuit:
(a) report it in housekeeping stream,
(b) turn the DM back on (up to a commandable number of times).

-13 Detectors with counts within the survey image that are less than a commandable level or greater than a commandable level will be turned off. This shall not cause a trigger.

3.5 RATE TRIGGER REQUIREMENTS

-1 There shall be an algorithm that scans the events rates (formulated in various energy intervals and detector array regions) looking for rate increases of more than a commandable level. These increases constitute a "trigger". The "BAT Trigger Algorithm" document by David Palmer will describe the details of this algorithm.

-2 The BSC's Trigger functionality shall build and maintain a table of trigger criteria capable of holding up to 500 criteria, each of which has at least the following parameters:

energy range (the 4 standard energy bins; note that the lowest edge of these bins is such that it is above the highest of the effective electronics threshold of all 32K pixels, i.e. the so called "software" threshold).
order of fit to background (0^{th}, 1^{st}, or 2^{nd} as a function of time).
duration and offset of background interval(s).
duration of trigger interval (4 msec to 32 sec) .
trigger interval phase factor (a fraction of the trigger interval).
detector region (e.g. quadrants, halves, and the whole array).
rate increase threshold level (sigma, sigma^2, percent change, or other).
exclusion of known sources using mask-tagged rates (a yes or no flag).
a set of 10 'merit' values that will be included in the message to the FOM (if the GRB is verified). Some of these merit values are fixed for the trigger, and some have their value assigned based on the properties of the burst. These values are single byte signed quantities (-127 to +127 in value). The negative range allows for the decrease of a burst's priority.
priority (typically 0-127), allowing throttling of cpu usage.

All of this is MinCoreReq.

-3 This table shall be updateable by ground command.
This table will be sent to the ground (SRR) every time it is changed.
This table will also be sent to the ground (SRR) every day.

-4 When one or more of the time-based appropriate trigger criteria is exceeded, the trigger criteria with the highest significance is the one used.

-5 Upon a successful trigger criteria, the BSC's Trigger functionality shall generate and send:
a) a Trigger Alert Message for transmission to TDRS The Trigger serial number, the Trigger Criteria number (i.e. table entry number), the significance of the detection (i.e. probably in sigma units), and about 20 other items (see the "Swift TDRSS Messages" document).
b) an APID packet into the BAT telemetry stream for the SSR,
c) an RT-RT Command to XRT and UVOT.
[There will be a "filter" (in the "engineering side of the flight s/w) that stops Alert Messages to TDRSS during Malindi downlink sessions. The triggers that will be stopped are those due to non-GRB origins. Those triggers due to real GRBs will only be delayed by a few seconds.]

-6 The Data Ingest Task continues to run (based on the R6K task scheduler) processing event data in case the Burst Response Task determines the trigger was due to a non-GRB.
If the trigger is determined to be non-GRB, then the Trigger Task resumes processing the data stream at the next place after the original trigger.

-7 The Trigger Task shall also alert the Burst Response Task for further generation and analysis of the internal data products. It is this highest-significance criteria that determines the time interval of the background and burst data detector maps to use, the energy range to use, and the detector array regions to use.

-8 The BSC shall not trigger an Alert as a result of event fluctuations caused by a s/c slew (i.e. triggering is disabled during a slew).

-9 The BSC shall be capable of executing 500 of these trigger criteria per second during normal background operations (i.e. the 34Kevt/sec times). Some criteria have to run multiple times within a given 1sec interval (e.g. those with a 4-msec timescale will run 250 times in that 1 sec) and some will execute slower than once per second (e.g. a trigger with a 32-sec timescale). The details how many criteria will be running in any given time interval are contained in the "BAT Trigger Algorithm" document by David Palmer.

-10 The BSC shall have the capability to adaptively decrease and increase the number of trigger criteria it actually executes. The criteria used to determine these increases and decreases may be based on (a) a cpu idle time monitor (provided by the engineering code) or (b) the amount of data pending processing by the Data Ingest Task. The adaptive process shall be stable against oscillations. The order of the adding & deleting trigger criteria from the active list shall be based on the priority parameter in the trigger criteria table.

-11 The rate trigger shall respond appropriately to changes in the detector active area, i.e. not triggering in response to count rate changes due to enabling or disabling of detectors, DMs, or Blocks, and for the various electronic pulser cycle activities (either in straight cal mode or in mixed mode). The method of insuring that that the trigger responds appropriately might be to disable all triggers that use the affected area. Disabling triggers will not affect the science goals for enabling/disabling of detectors, DMs, or blocks since that is done rarely. Science goals will be affected if excessive calibration deadtime causes statistically significant changes in the count rate. There is no requirement for the BSC to correct for calibration deadtime; affected triggers will likely be disabled.

-12 Excessive triggering will be prevented by the use of adjustable parameters within the trigger code.

3.6 IMAGING TRIGGER REQUIREMENTS

-1 There shall be an algorithm that periodically (a) produces sky images of the FOV, (b) searches for new unknown sources, and (c) searches for sources in the known-sources table which are outside of their expected variability range.

-2 There shall be a table of trigger criteria controlling this algorithm. This table shall hold at least 16 criteria. Each trigger criteria contains, at a minimum, the following parameters:
The energy range (which combination of the 4 coarse E-bins).
The frequency of execution (e.g. 64 sec, 5 minutes, one orbit's observation).
The expected-flux threshold for known sources to be used in the non-iterative CLEAN algorithm.
The threshold for new sources, specified in image SNR units.
The classification (GRB or transient) assigned to new sources. (This is passed to the Burst Response Task.)
A priority (typically 0-127) used for throttling the CPU usage.
All of this is MinCoreReq except that the number of simultaneously running triggers is 1.

-3 This table shall be updateable by ground command.
This table will be sent to the ground (SRR) every time it is changed.
This table will also be sent to the ground (SRR) every day.

-4 The RAD6000 shall pass a detector map (accumulated by the Data Ingest Task) and the list of the known sources in the field of view that exceed the specified expected-flux threshold, to the DSP for imaging (FFT, mask convolution, & FFT^{-1}).

-5 The imaging will use a non-iterative CLEAN algorithm, based on the detector position in the array and within its module, and on the set of known sources. The CLEANed detector map will be imaged (using the FFT algorithm).

-6 The FFT image shall be searched for sources. A list of sources and their strengths shall be passed back to the RAD6000.

-7 The RAD6000 shall select the sources that are either (a) unknown or (b) outside of their expected flux range, and act accordingly. These shall constitute Image Triggers. (For reference, Image Triggers are separate and distinct from Rate Triggers.)

-8 For each unknown source, the RAD6000 shall instruct the DSP to produce a fine image (back projection method), and return the exact location and strength.

-9 For Imaging Triggers generated from trigger timescales less than 90 sec (a commandable value), the new source shall be classified as a GRB for processing and discussion purposes. The RAD6000 shall (a) notify the Burst Response task with this information, and (b) send a Burst Alert Message and a Burst Position Message to TDRSS, to the FOM, and to the SRR.

-10 For Imaging Triggers generated from trigger timescales longer than 90 sec (same commandable value), the new source shall be classified as a Transient for processing and discussion purposes. The RAD6000 shall (a) not notify the Burst Response task, and (b) send a Transient Alert Message and a Transient Position Messages to TDRSS, to the FOM, and to the SRR. For reference, the Transient Position Message also contains the detector map used in the detection of this transient.

-11 For each known source outside of its expected intensity range, the Imaging Trigger shall alert the FOM of a known-transient source with the appropriate information.

-12 The Imaging Trigger shall be capable of running one trigger criterion per survey image under normal background conditions (i.e. the 34 K events/s event rate).

-13 During activities in support of the Imaging Trigger, the DSP shall be interruptible in response to the Burst Response to a Rate Trigger within 250 ms. If the Imaging Trigger DSP operations are interrupted, the Imaging Trigger for that criterion for that time-step shall be aborted and not restarted. (This is done for simplicity.)

3.7 BURST RESPONSE REQUIREMENTS

-1 If a Rate Trigger or a short timescale Image Trigger is detected, the BSC shall construct a sky image from the detector map. (Note: this processing will be done in the DSP with the FFT and back-projection algorithms, based on (a) the background-subtracted detector map produced by the RAD6000 and corresponding to the highest SNR Rate Trigger, or (b) the non-background-subtracted detector map if initiated by an Image Trigger.)

-2 From this sky image, the BSC shall determine the peak location of the GRB. (Note: this determination will be done in the DSP.) The time between the initial trigger detection and the availability burst/no_burst result and the burst position to the FOM shall be less than 6 sec (1.5sec expected; see the MRD Table EB1). {For reference, the total time interval is 6.2 sec from trigger detection to arrival at the s/c.)

The peaks from the locations of known sources will be ignored (should there still be residuals from a less-then-perfect background subtraction). The counts in the largest remaining peak will be tested by comparison with counts derived from the rate trigger.

-3 The Burst Response Task shall convert the peak position from BAT instrument coordinates (e.g. theta,phi) into celestial coordinates (RA,Dec). The rotation matrix will be provided to the BSC by the engineering code.

-4 The Burst Response Task shall store to the SSR a complete description of the criteria that caused the trigger.
The Burst Response Task shall alert the Survey Task to generate a series of 1-minute detector rate maps for inclusion in the SRR stream.
The Burst Response Task shall alert the Data Ingest Task to generate mask-tagged lightcurve series for the peak position for inclusion in the SRR stream.

-5 If a GRB is determined to have occurred, the Burst Response element shall send a Burst Position Message for transmission to TDRS. A Burst Observation Request shall be transmitted to the FOM which contains (a) the burst position in both coordinate systems, (b) the Target Number (which is the same as the BAT Trigger Number), (c) the 10 parameters for the merit calculation, and (d) other items yet to be specified.

-6 If a GRB is determined to have occurred, a 10-min event history shall be stored in the SRR and transmitted to the ground. The BSC shall implement a command to define the distribution of event data preceding and following the burst to include in the downlink. The BSC shall implement a command to store and transmit non-burst related event-by-event data to the ground. This command will be used for diagnostic/debug purposes only.

-7 If a GRB is detected, the BSC shall transmit the detector count rate (i.e. light curve) to TDRSS. The full lightcurve will likely be broken up into 3 separate Messages to TDRSS. The entire light curve will be available on the ground within 200 sec. A copy of the lightcurve is also sent to the s/c for the SRR.

-8 If a GRB is determined to have occurred, a mask-tagged rate for the burst position will be set up in the Data Ingest task.

-9 At the end of the 10 minutes of event-by-event data, the sample interval for survey data will be reduced to a lower value (~1 minute) until the automated program returns to the uploaded observing program. (Note that the "10 minute" interval shall be a commandable value.)

-10 If the result of the image is that there was no GRB (i.e. no point source or that it was due to a less-than-threshold fluctuation in a known source), then the Burst Response element shall send a Burst Position Message with the "non-GRB" flag set to TDRS and the NFIs.
It shall also insert the detector rate map into the SSR data stream.

-11 The Burst Response Task shall accept targets (pre-planned, uploaded TOOs (forwarded from the FOM), etc.) and act on them as if a regular trigger criteria was satisfied (this will be the entry point for the TOO commands into the BAT operations procedure.)

-12 The Burst Response Task shall receive a copy of the "next target info" message (from the FOM) so that BAT can generate the appropriate burst-type data products or the survey-type data products based on the mode/configuration bits/bytes in the Target Message.

-13 The Mode/Configuration determine how BAT operates during a data-taking session (especially during a new target acquisition: burst, TOO, survey, etc.).
The modes and configurations are:
Survey integration interval (e.g. N minutes each, 1-10 min)

Burst pre-trigger time interval (0-10 minutes)
Burst post-trigger time interval (0-10 minutes)
Burst detector map integration interval (e.g. N minutes each, 1-10 min)

-14 All of this is MinCoreReq (except 3.7.9).
This table is a summary of data products produced by the "science code", It is provided here for reference and perspective, and is not a requirement (and therefore not subject to verification and validation testing).

Data Product	**Destination**	**Range of Data Size**	**Frequency**
Trigger Alert	TDRSS/FOM	1-2KBytes	1-4 / day
Burst Result	TDRSS/FOM	1-2KBytes	1-4 / day
Light Curves	TDRSS/FOM	1 KBytes	3 / GRB
Event-by-event data	Normal D/L	1-10 MB	1-10 / GRB
Hi-time resolution lightcurve	Normal D/L	TBD KB	1 / GRB
1-min survey maps/images	Normal D/L	TBD KB	TBD / GRB

3.8 SURVEY REQUIREMENTS

-1 The time interval for the accumulation of the survey maps shall be commandable over a range of 1 to 10 minutes with 1 minute quantization.

-2 The survey maps consist of 32K spectra with 80 E-bins. The number of E-bins is a adjustable number, as well as the edges and widths of each bin. And for inclusion into the telemetry stream, the depth (i.e. the number of bits used for the counts in each bin) is commandable.

-3 The BSC's Survey element shall maintain the following data products and transmit them to the ground according to the table below. The "size" and "frequency" columns are provided for reference (indicative of the order of magnitude) and are not requirements themselves.

Data Product	**Destination**	**Range of Data Size**	**Frequency**
Detector Light Curves	Normal D/L	200 bytes/second	continuous
Masked Light Curves	Normal D/L	200 bytes/second	continuous
Survey Detector Maps	Normal D/L	1-2 MBytes	1 / 5-mins
Pulsar Data	Normal D/L	20 KBytes	1 / 5-mins
GRB Events Lists	Normal D/L	~ 48 MBytes	1 / GRB

-4 The Survey element takes the data products produced by the Data Ingest task (i.e. histogram accumulations, detector array count rate maps, etc.) and (a) determines which "files" they are written to RAM Disk, (b) determines what packet segmentation is needed, and (c) determines what priorities are assigned to transmit the data products across the 1553 bus to the s/c and SRR.

-5 The Survey shall immediately stop the generation of the "5-minute" maps upon receipt of a Trigger (Rate or Image) and from receipt of a Slew Message (from the spacecraft).
The Survey shall resume to formation of the "5-minute" maps when the Burst Response Task has completed its set of data products for the current new burst.

-6 All of this is MinCoreReq.

3.9 CALIBRATION REQUIREMENTS

There are 3 types of calibration activity: (a) electronic pulser events to check the offset, gain and linearity, (b) tagged-source events to monitor the absolute energy calibration of the detectors and electronics and the absolute effective area calibration, and (c) XA1 threshold monitoring. Only the "gain and offset" e-pulser function and the tagged-source spectral accumulation are MinCoreReq; and only those two will be subject to validation and verification procedures. The other functions listed in this sub-section are provided for reference and are not subject to validation and verification.

The offset, gain, & linearity calibrations can be broken up into a set of nested cycles with increasing time intervals (respectively) based on the required knowledge each type of calibration information. The times at which the various cal cycles are executed are based on either a schedule of periodic commandable cycles or on asynchronous events (e.g. slews, …). Alternatively (and equally acceptable), these non-MinCoreReq functions can be accomplished by explicit individual ground commands (administered by ground operations) and by event-by-event capture and download.

3.9.1 General Calibration Requirements.

-1 The BSC shall conduct calibration activity for the purpose of calibrating the XA1 electronics. The schedule of calibration activities (i.e. cycles, time intervals, and which Blocks) shall be contained in a table.
This table will be sent to the ground (SRR) every time it is changed.
This table will also be sent to the ground (SRR) every day.

-2 In the normal operating mode 0, 1, or N Blocks at a time shall be dedicated to "calibration" mode or "mixed" mode. That is, all remaining Blocks will be dedicated to science gathering. This calibration activity shall only be interrupted during a GRB science data gathering session (i.e. the post-trigger part of the "10 min" interval) .

-3 The number of pulses injected by the pulser electronics and the pulser rate shall be programmable.

-4 The pulser DAC level shall be programmable at the DM level.

-5 There shall be 7 basic electronic calibration cycles through which the BSC calibration functionality will loop:
Offset cycle
Segment & Strip slope & offset
Gain & Offset [MinCoreReq]
Full Linearity
Differential Non-linearity
Threshold Monitor
Noise Floor Monitor
Only the Gain & Offset function is MinCoreReq.

-6 If the scheduling of the cycles is in the periodic method, then the 5 electronics calibration cycles shall run on commandable automatic time intervals.
The intervals for each cycle shall be such that they are each an integral multiple of the shortest cycle (i.e. the Offset cycle).

-7 Each cycle shall have a master dis/enable command control.

-8 Each cycle shall be executable by direct manual command (bypassing the automatic cycling interval).

-9 The offset (both in raw adu's and possibly in keV units) shall be maintained for each of the 32K detector elements (for use by the Data Ingest Task).

-10 For each command to the BCDH that initiates calibration pulses (The "SetMode" command), the BSC will calculate means, sigma's, and number of outliers for each affected DM. These will be reported to the ground upon completion. For the short format, the BSC will find the mean/sigma/outliers for the energy. For the long format, the BSC will find the mean/sigma/outliers for the energy, strip ADC, and segment ADC. The testing of the long format calibrations prior to the use of the actual hardware will involve simulations that do not produce realistic values or variations.

-11 The changes of the parameters (both in software tables and hardware DAC levels) shall happen at the boundaries of the survey map accumulations (typically every 5 minutes).

3.9.2 Offset Cycle requirements.

-1 For this cycle, the BSC shall configure the pulser to output a commandable number of pulses (1-1023, typically 100 pulses) at the commanded rate (500 - 1000, typically 660 events/sec) at a commandable level into each of the commanded channels.

-2 The BSC shall gather the results of this cycle (mean and sigma of the pulser peak) and report them to the ground for evaluation.

-3 The BSC shall take the results of the offset calculation to adjust the appropriate Vc_ref DAC to keep the XA1 operating within the linear range of the device. (These changes will typically be of order 0.5(TBC) keV which is small compared to the intrinsic energy resolution of the detectors, and therefore will not introduce jump discontinuities in the data stream which will effect the other operations of the BSC or ground analysis.)

-4 The BSC shall loop through each DM every X minutes (where X is a commandable interval of order 1-60 minutes with a quantization of 0.5 minute) for this calibration cycle.

-5 This is non-MinCoreReq.

3.9.3 Segment and Strip Slope & Offset Requirements

-1 For this cycle, the BSC shall configure the pulser to output a commandable number of pulses (1-1023, typically 100 pulses) at the commanded rate (500 - 1000, of order 600 events/sec) at a commandable level into each of the commanded channels.

-2 For this cycle, the BSC shall place the DM under calibration in the Calibration Long Event format mode.

-3 The BSC shall calculate the means of the Segment & Strip pulse height peaks, make a linear fit to the Segment-peak means and the Strip-peak means and find the Slope and Offset for each. [We may find that using only the lowest and highest of the 4 or 32 peaks is sufficient to determine the 4 coefficients, in which case the number of channels pulsed will be reduced and the coefficient calculation shortened appropriately.] These four numbers shall then be downloaded into each DM and transmitted to the ground.

-4 The BSC shall loop through each DM every M multiples of the Offset-only cycle, where M is a commandable number (and translates into a rep-rate of at least every hour for this calibration cycle).

-5 The BSC shall gather the results of this cycle (the 4 coefficients) and report them to the ground every hour for evaluation.

-6 This is non-MinCoreReq.

3.9.4 Gain & Offset Requirements

-1 For this cycle, the BSC shall configure the pulser to output a commandable number of pulses (1-1023, typically 100 pulses) at the commanded rate (500 - 1000, typically 660 eventst/sec) at a commandable level into each of the commanded channels and then at a second commandable level.

-2 The BSC shall calculate the mean & sigma of these to pulser level peaks and report them to the ground for evaluation.

-3 The BSC shall loop through each DM every N seconds where N is a commandable, where N is a commandable number (and translates into a rep-rate of at least once every hour for this calibration cycle).

-4 This is MinCoreReq.

3.9.5 Full Linearity Requirements

-1 For this cycle, the BSC shall configure the pulser to output 8 pulses at a 8 different commandable levels into each of the commanded channels.

-2 The BSC shall gather the results of this cycle (means [and optionally the sigma's] of the pulser peaks) and report them to the ground for evaluation.

-3 The BSC shall loop through each DM every O multiples of the Offset-only cycle, where O is a commandable number (and translates into a rep-rate of at least once every day for this calibration cycle).

-4 At a rate slower than the basic rate, the full histograms will be reported to the ground (to allow for more detailed analysis of the shape of the peaks, the fliers, and other analyses).

-5 This is non-MinCoreReq.

3.9.6 Differential Non-Linearity Requirements

-1 For this cycle, the BSC shall configure the pulser to output a series of closely spaced levels such that the resulting pulser spectrum has all the peaks highly overlapped (the peak spacing is much smaller than the intrinsic peak width).
The number of pulses per level shall be a commandable value.
Whether this is done to each of the 128 channels in each XA1 or just to a single representative channel of each XA1 is an open question.

-2 The BSC shall gather the results of this cycle (a spectrum) and report them to the ground for evaluation.

-3 The BSC shall loop through each DM every P multiples of the Offset-only cycle, where P is a commandable number (and translates into a rep-rate of at least once every week (or maybe every month) for this calibration cycle).

-4 This is non-MinCoreReq.

3.9.7 Threshold Measurement Requirements

-1 For this cycle, the BSC shall configure the pulser to output a series of pulses (commandable from 1-1023, typically 1000) slightly above the threshold level of the DM. Then the level of the pulses shall be decreased and the XA1 discriminator firing percentage information gathered.

-2 The BSC shall gather the results of this cycle and report them to the ground for evaluation. The results are the probability of firing at a set of pre-defined levels.]

-3 The BSC shall loop through each DM every Q multiples of the Offset-only cycle, where Q is a commandable number (and translates into a rep-rate of at least once every day).

-4 The specified DMs and/or Blocks shall have to be removed from the "active" array when undergoing this threshold procedure.

-5 This is non-MinCoreReq.

3.9.8 Noise Floor Measurement Requirements

-1 This cycle shall provide a mechanism to measure the "noise floor" of each channel on each XA1 ASIC and for each ASIC as a whole (where "as a whole" is defined to be only those "good channels" are enabled). This noise floor level is defined as the threshold level at which a given channel produces five (5) times the average detector background rate (typically about 5*0.5 cnts/sec).

-2 For the individual-channel cycle, a single channel within an XA1 ASIC shall be enabled. The basic technique is to start at some initial threshold level, record the event rate from that channel, lower the threshold by some commandable increment, record the event rate, and continue this cycle until the 5x-background_rate is achieved. [An alternative scheme would be to start at the low end, increasing the threshold until the event rate drops below the commandable rate criteria.]

-3 For the whole-asic cycle, the same basic technique of lowering (or raising) the threshold level for the ASIC with all "good channels" enabled, and monitoring the event rate per ASIC. The threshold level shall be determined when the event rate reaches the value N_chan_enabled*5*0.5 evts/sec.

-4 Both of these procedures will be done with the HV on (at some commandable value) and off. For the HV-on case, the nominal background rate will have to be adjusted to compensate for any strong sources in the FOV.

-5 The inclusion of these events from this procedure in the telemetry stream shall be controlled by a commandable dis/enable flag.

-6 The BSC shall loop through each DM every Z multiples of the Offset-only cycle, where Z is a commandable number (and translates into a rep-rate of at least once every month).

-7 This is non-MinCoreReq.

3.9.9 Tagged Source Requirements

-1 The tagged source spectrum shall be analyzed for each detector each commandable period (~1 hour).

-2 A background-subtracted peak area (counts/sec), centroid value and width (raw pulse height units) shall be calculated for each spectrum.

-3 They shall be sent to the ground in a separate telemetry packet.

-4 This is MinCoreReq.

Calibration Summary

-1 This table is a summary of calibration data products produced by the "science code", It is provided here for reference and perspective, and is not a requirement (and therefore not subject to verification and validation testing).

Item	Size (Bytes)	Rep Rate	Bytes/sec
1a) Average DM Offset	512-1K	5 min	1.7 – 3.4
b) DM Offset of each channel	64K-128K	5 min	218 – 437
c) DM Offset Pulser events	13 M	1 week	21.7
2a) Chan Slope & Offset	2K	1 hour	0.57
b) Chan Slope & Offset events	26 M	1 week	43.3
3) Pixel Slope & Offset	128K	1 hour	36.4
4) Pixel Linearity Run	512K	1 day	6.07
5) Differential Non-Linearity Run	100*32K	1 month	1.35
6) Threshold Level	10*32K	1 day	3.53
7) Noise Floor Monitor	32K	1 month	0.02
8) Tagged source, area, peak, FWHM	128K	1 hour	35.5

3.10 DIAGNOSTIC FUNCTIONS AND MODES

-1 The BSC will support the inclusion of calibration pulses in the event-by-event data. With event turned off, this could be used for all the calibrations listed in section 3.9.

-2 The BSC will support a command to initiate burst mode. In that way, one can get about 10 minutes of event-by-event data for diagnostic purposes.

-.3 The BSC will support passing all housekeeping to the ground for short periods (typically 10 minutes) for diagnostic purposes.

4 BAT C&DH FLIGHT SOFTWARE REQUIREMENTS

This section, the largest of the document, specifies all the general requirements dictating the design and implementation of the BAT C&DH software for Swift. The following subsections detail the software's functional and resource requirements as well as those imposed on the development of the software itself.

The following areas are covered in this section: System Design Criteria, Initialization, System Management, Command Processing, Telemetry Processing, Instrument Time Management, Instrument Safing, Onboard Software Maintenance, Power Management and Hardware Interfaces.

4.1 GENERAL

-1 The format of all telemetry and commands shall be in accordance with CCSDS standards and implementation.

-2 All commands executed by the instrument shall be verifiable on the ground. Where end point telemetry verification is not possible, separate counters in software shall increment to indicate whether the command was processed or rejected by the instrument flight software.

-3 The instrument shall be capable of sustaining itself in the normal operating mode, without ground communication or intervention, for 72 hours. Instrument health and safety shall not require routine FOT involvement.

4.2 HARDWARE INTERFACE REQUIREMENTS

As can be seen in the system diagram in section 2, the flight software has many hardware interfaces with which it must communicate. This section addresses flight software requirements related to interfaces with hardware devices. These include hardware devices on the processor itself, such as the QHSS, as well as external devices interfacing with the processor over the 1553B bus or over the local PCI bus in the Instrument Processor. The majority of the requirements listed below are *high-level* functional requirements. The *low-level* implementation details for these interfaces can be found in the appropriate Interface Control Documents. In case of conflicts between the ICD and this document, the ICD shall take precedence.

4.2.1 BAT IPE External Hardware Interface Requirements

4.2.1.1 1553 Bus Spacecraft Interface

-1 The BAT software shall interface for commands and telemetry to the spacecraft via the MIL-STB-1553B data bus. The interface is documented in the Swift 1553 Bus Protocol Interface Control Document.

4.2.1.2 Power Box

-1 The flight software shall interface with the BAT power box as defined in the Swift BAT IPE to BAT Power Box Interface Control Document.

4.2.1.3 BAT IPE to BCDH interface

-1 The flight software shall interface with the Block Command and Data Handlers as defined in Swift BAT BCDH Software ICD and the BAT DMC Specification.

4.2.2 BAT IPE Internal Hardware Interface Requirements

The BAT Instrument Processing Electronics contains the RAD6000 instrument processor, 1553/DRAM/EEPROM card, the 21020 DSP card, the Multi-channel interface card, and the low voltage power supply (LVPC) card.

-1 The flight software shall support the following PCI transfers at 25MHz:
   - Memory reads and writes
   - I/O reads and writes
   - PCI slave to slave transfers
   - PCI Burst Mode reads and writes
   - DMA transfers to PCI slaves
   - RAD6000 local memory transfers to PCI slaves
   - PCI configuration transfer

4.2.2.1 RAD6000 Processor Card

The RAD6000 processor card runs at 25MHz (about 27MIPS). It contains 7Mb of local RAM, 1Mb of EEPROM non-writable in flight, and 64kb of startup ROM. All memory is radiation hard and EDAC protected. The primary external interface for the processor is the PCI bus. The RAD6000 card is a PCI bus master. All communication with the other cards in the BAT IPE occurs over the PCI bus. The processor card also contains a Quad High-speed Serial (QHSS) interface chip. The interface to the QHSS is also over the PCI bus.

-1 The flight software shall communicate with the RAD6000 processor card as defined in the "Software User's Guide for the RAD6000 Processor" (Lockheed Martin document #204A496).

-2 The flight software shall be capable of configuring and utilizing the processor’s UART serial interface for a host to target connection using the VxWorks raw serial protocol.

-3 The RAD6000 processor's Quad High Speed Serial (QHSS) will not be used for this mission. The flight software shall initialize and configure this interface hardware to a known safe state.

-4 The flight software shall be capable of configuring and utilizing the processor’s Local Input Output (LIO) chip as required. The specifics for using this chip shall be as defined in the "Critical Item Development Specification for RAD6000SC Local Input/Output (LIO) Chip" " (Lockheed Martin document).

4.2.2.2 21020 DSP Card

The 21020 DSP card interfaces to the RAD6000 via the on-card PCI Target/DMA controller. Details of the DSP engineering code requirements are found in section 4.13.

-1 The flight software shall communicate with the 21020 DSP card over the PCI bus.

4.2.2.3 1553/DRAM/EEPROM Card

The 1553/DRAM/EEPROM card has 256 Mbytes usable DRAM with onboard EDAC capability (320MB total). The card implements fast page mode transfers as Target/DMA controller. The card has a 1553 B interface using UTMC 1553 Summit chip. The card has 1 Mbyte EEPROM for normal mode code storage

-1 The flight software shall communicate with the 1553/DRAM/EEPROM Card over the PCI bus.

4.2.2.4 Multi-Channel Interface Card

The Multi-channel interface card provides 16 individual IEEE 1355 compliant protocol interfaces to receive and transmit data to/from the 16 block controllers. One interface is provided to each BAT block command and data handler (BCDH). The Multi-channel card buffers science event and housekeeping data received from the instrument BCDHs. The instrument data is DMA transferred via the PCI Target/DMA controller to the DRAM card. The Multi-channel I/F card inserts markers in the bulk memory data to allow differentiation of the data from the 16 BCDHs.

The Multi-channel I/F card buffers command data to be sent to each of the 16 BCDH controllers and provides registers accessible via the PCI backplane for command and control data from the RAD6000 processor.

The Multi-channel I/F card provides 32 MHz clock signals to the 16 BCDHs, and forwards the spacecraft 1 Hz clock signal to the BCDHs.

-1 The flight software shall communicate with the Multi-channel Interface Card over the PCI bus.

4.3 C&DH SOFTWARE INTERFACES

4.3.1 <u>Science Code Interface</u>

-1 The flight software shall provide services and communicate with the BAT Science Code as documented in the Swift BAT C&DH Software to BAT Science Code ICD.

4.3.2 <u>FoM Interface</u>

-1 The flight software shall provide services and communicate with the FoM Code as documented in the Swift BAT C&DH Software to BAT Science Code ICD.

4.4 INITIALIZATION

This section discusses requirements related to the initialization of the flight software. The subsections list general requirements and those for both cold and warm restarts of the processor.

4.4.1 System Initialization

-1 The system startup software shall initialize the processor and its on-board devices and peripherals as needed for its own operation.

-2 The system startup software shall zero processor RAM if it detects power was cycled.

-3 The flight software shall begin servicing the hardware watchdogs, after either a warm or cold restart, in less than 3.0 seconds.

4.4.2 Cold Restarts

-1 The flight software shall be able to perform a cold start in which all code and data areas are re-initialized and all software tasks are restarted.

-2 The flight software shall perform a cold restart as a result of the following:
processor local watchdog timeout
too many consecutive warm restarts

-3 The flight software shall perform a cold restart in response to a ground command.

-4 The flight software shall be able to autonomously perform a cold restart in response to certain detected anomalies or single event upsets (SEU).

-5 Each cold restart, along with its cause and any relevant data shall be stored for downlink to the ground.

-6 After a cold reset, the flight software shall automatically start a special stored command script. This script can be used to configure the software as desired after a reset.

4.4.3 Warm Restarts

-1 The flight software shall be able to perform a warm restart in an attempt to recover from a detected software anomaly with minimal impact on normal processing activity.

-2 During a warm restart, the system startup software shall stop and restart all tasks.

-3 During a warm restart, the system startup software shall not reinitialize the RAM data areas associated with each task. After a warm restart, all software tasks shall attempt to continue processing from the same state they were in prior to the warm restart.

-4 The following normal mode functions shall automatically resume after a warm restart if they were executing prior to the warm restart:
real-time command ingest
onboard state of health monitoring
real-time telemetry output
housekeeping and event generation
bulk memory data storage and playback
BCDH commanding and data capture
science data processing
detector calibration activity

-5 The flight software shall perform a warm restart in response to a ground command.

-6 The flight software shall be able to autonomously perform a warm restart in response to certain detected anomalies or single event upsets.

-7 Each warm restart, along with its cause and any relevant data, shall be stored for downlink to the ground.

-8 The flight software shall maintain a count of the number of warm restarts that have occurred since the last cold restart. This counter shall be available in housekeeping. The flight software shall reset the warm restart count by ground command.

-9 If the FSW exceeds a specified *maximum number of warm restarts*, the flight software shall initiate a cold restart by allowing the processor watchdog timer to expire. The value of this number shall be selectable by ground command.

-10 After a warm reset, the flight shall not start the special stored command script containing initial configuration commands.

4.5 SYSTEM MANAGEMENT

This section lists those requirements related to the system-level management of the flight software. The requirements for system operating modes, task-to-task communications, system activity scheduling and functions to be performed by the operating system are all discussed in this section.

4.5.1 Operating Modes

This section lists those requirements associated with the flight software operating modes. The flight software will have two basic operating modes: launch mode and enhanced mode. Launch mode is intended to provide all instrument functionality given the current understanding of the software requirements at the time of spacecraft launch. Enhanced mode allows for in-flight modifications to the software. This provides the ability to support new requirements or fix problems discovered after launch.

-1 After a cold restart, the flight software shall begin executing in launch mode.

-2 The launch mode code, stored in PROM, shall be hardware write disabled. It shall not be possible to write to launch mode code on-orbit.

-3 Execution of the launch mode code shall not depend on any code or data stored in the enhanced mode (writable) section of EEPROM.

-4 Transition from launch mode to enhanced mode shall be by command only.

-5 The enhanced mode code shall be stored in EEPROM. It shall be possible to update the contents of the enhanced mode EEPROM while in flight.

4.5.2 Task-to-Task Communications

-1 The flight software shall provide services for routing task-to-task messages. These services shall support both the one-to-one and one-to-many routing schemes.

-2 The task-to-task communication services shall be designed such that each path of communication is independent of the others. It shall not be possible for an error condition on one path to shut down communications across another path.

-3 The format of task-to-task messages shall be in accordance with CCSDS packets.

-4 The flight software shall maintain counters for recording task-to-task communication errors. These error counters shall be available in nominal housekeeping telemetry.

-5 The flight software shall be capable of dynamically enabling/disabling and otherwise change the routing of task-to-task communication paths in response to ground commands.

4.5.3 Error Reporting

-1 The flight software shall provide a common mechanism for reporting and counting errors and significant events.

-2 Each error condition shall have an associated event message, which shall be a CCSDS telemetry packet containing pertinent information as to the nature of the error condition.

4.5.4 System Activity Scheduling

-1 The flight software shall schedule all predictable system activity. This scheduling scheme shall support a minor frame resolution of at least 10ms and a major frame resolution of one second.

-2 The scheduling activities shall be defined using a table. Each minor frame shall be capable of scheduling at least three different activities.

-3 The flight software shall be capable of scheduling of activities over the internal and external communication buses (except the 1553 bus*). The flight software shall be capable of scheduling activities within and between individual software components.

*1553 bus activity must be scheduled by the bus controller, which on this mission is the spacecraft computer.

-4 The flight software shall have the capability to disable or enable specific activities defined in the system schedule table.

-5 The flight software shall continue to operate in the absence of the spacecraft 1Hz clock.

-6 The flight software shall continue to if the spacecraft 1Hz clock becomes noisy (firing much more often than 1Hz).

4.5.5 Operating System Functions

-1 The operating system shall provide a common set of mechanisms necessary to support real time systems such as multitasking support, CPU scheduling, basic communications and memory management.

-2 The operating system shall provide multitasking capabilities using an event-driven, pre-emptive, priority based scheduler.

-3 A task's priority and the availability of resources required for its execution shall govern task execution.

-4 The operating system shall provide support for inter-task communication and synchronization.

-5 The operating system shall provide real-time clock support.

-6 Task definitions and attributes shall be maintained in a system table or file.

4.6 COMMAND PROCESSING

This section describes those requirements related to flight software command processing. This includes ground commands received via the 1553 interface as well as autonomous command scripts.

4.6.1 Ground Commands

-1 The flight software shall accept real-time commands up to 62 bytes in length from the spacecraft at a rate of 20Hz.

-2 The flight software shall accept commands, which are formatted to comply with the CCSDS Telecommand Standards, Blue Book Issue 2.

-3 The flight software shall accept either segmented or unsegmented commands as indicated by the CCSDS segmentation flags. For segmented commands, the software shall re-combine the packet segments before executing the command.

-4 The flight software shall support variable length CCSDS command packets. The maximum command length shall be 256 bytes (after segment re-construction) and all commands shall be an even number of bytes.

-5 The flight software shall execute, via ground command request, a NOOP command that does nothing except increment the command sequence counter.

-6 The flight software shall telemeter status and discard the current real-time command packet if any of the following errors occur:
    packet checksum fails validation
    an invalid packet length is detected
    an invalid ApID is detected
    an invalid or out-of-sequence packet segment

-7 The flight software shall route individual CCSDS command packets to their destination based solely on the packet's AppID.

-8 The following statistics, as a minimum, shall be maintained on command ingest performance:
    current command packet count
    command packets accepted
    command packets rejected

4.6.2 Command Scripts

The following sections list the flight software requirements for processing stored command scripts. Command scripts provide an on-board method for initiating instrument commands based on data or time variables.

-1 The flight software shall accept Stored Command Scripts from the command uplink interface. The flight software shall be capable of supporting at least 64 SCSs.

-2 The flight software shall accept sequences of commands, which will execute based on an absolute time. Each command in the sequence shall be able to support a different execution time.

-3 The resolution for absolute time tagged commands shall be at least one second.

-4 Absolute time tagged commands shall be issued to their destination when the spacecraft time matches the individual commands absolute time tag.

-5 The flight software shall accept sequences of commands that will execute based on a relative time.

-6 The resolution for the time tag of a relative time tagged command shall be one second. A command's time tag shall be interpreted as the execution delay from the previous command in the sequence. A relative time tag of zero shall cause the command to be issued as soon as possible after, and in the same second as, the previous command.

-7 The FSW shall support loading and dumping of stored command sequences.

-8 The flight software shall allow additional groups of stored commands to be loaded to the instrument. These additional commands shall be executed in parallel with current command loads.

-9 The execution of an SCS shall begin in response to a command. Once started, the flight software shall process the SCS autonomously.

-10 The flight software shall allow at least 10 SCSs to execute concurrently.

-11 The flight software shall support the following stored command sequence control requests from a ground command or from within another SCS:

Start a specified stored command sequence
Stop a specified stored command sequence
Disable a specified stored command sequence
Enable a specified stored command sequence

-12 It shall not be possible to start a stored command sequence unless the SCS is in the *enabled* state. If a request to start an SCS is received while the SCS is already running, the software shall reject the new request and allow the SCS to continue.

-13 An active stored command sequence shall completely finish processing if it is *disabled* while executing. However, once complete it shall not be possible to restart the sequence without first re-enabling it.

-14 Once enabled, it shall be possible to restart a stored command sequence without having to re-enable it between executions.

-15 The FSW shall provide a programmable default enable/disable state for each SCS

-16 The flight software shall generate an event and discard the stored command if one of the following errors occur:

command checksum fails validation
invalid packet length is detected

invalid packet AppID is detected

-17 The flight software shall maintain status for each stored command sequence including:
Sequence status (enabled/disabled and running/idle)
Total # of commands executed for all sequences combined
Total # of command errors for all sequences combined

4.6.2.1 Stored Command Priorities and Timing

-1 The flight software shall be capable of time tagging stored commands in at least a 1 second increments. The flight software shall be capable of issuing up to a maximum of 10 commands per second.

-2 Relative time tagged commands shall be issued within 100ms of their programmed execution time.

-3 Absolute time tagged commands shall be issued with a time variance of no more than 100ms.

-4 The flight software shall provide a means of prioritizing stored commands in the event of an execution scheduling conflict. Stored commands sequences shall have at least 2 distinct priority levels.

4.7 TELEMETRY PROCESSING

The requirements in this section deal with the collection, storage and output of telemetry data.

4.7.1 <u>Telemetry Collection</u>

-1 The flight software shall be capable accepting telemetry data from any source within the BAT, and formatting it for output to the spacecraft over the 1553 bus.

-2 The flight software shall be able to selectively discard outgoing data according to a filtering scheme controllable by ground commands. This filtering scheme shall control how often a particular packet is sent to the spacecraft.

-3 The telemetry filtering scheme shall allow a particular packet to be sent every time it is generated, sent every Nth time it is generated, or never sent.

4.7.2 <u>Telemetry Format Specifications</u>

The following sections discuss the requirements of the format specifications for telemetry data.

-1 The telemetry packets collected by the flight software and downlinked to the ground shall be in the format of CCSDS packets.

-2 CCSDS telemetry packets shall consist of a primary header, a secondary header and, when necessary, the associated application data.

-3 The CCSDS secondary header shall be a time code with the most significant bit set to zero to indicate the packet is a non-standard CCSDS secondary header.

-4 CCSDS telemetry packets shall be capable of containing variable length data fields.

-5 The flight software shall do nothing to restrict the segmentation of telemetry packets as defined by the CCSDS standard.

-6 The flight software shall support telemetry packet lengths in the range of 8 to 958 bytes. *(This length represents the entire size of the packet including the primary header, secondary header and the application data.)*

-7 The format and structure of the primary and secondary packet headers shall be as defined in CCSDS Packet Telemetry Standards, Blue Book Issue 2. The values of the individual fields in the packet headers shall be as defined in the Swift T&C Handbook, Volume 1.

-8 The last two bytes of every telemetry packet shall contain a packet checksum, calculated and filled in just before the packet is placed in the telemetry output stream.

4.7.3 Telemetry Output

-1 After initialization, the flight software automatically begin output of telemetry to the spacecraft.

-2 The flight software shall be able to stop and re-start telemetry output on command.

-3 The flight software shall maintain status and statistics on the real-time output of telemetry data. This status shall be reset-able by ground command and available in instrument housekeeping data.

-4 The flight software shall maintain a buffer containing a select group of various data types each containing different bit patterns. This buffer shall be sent to the ground in a CCSDS packet when requested.

-5 The flight software shall be able to output telemetry packets in any of the ApID ranges corresponding to the following telemetry types:

Real-time telemetry
Stored telemetry
TDRSS messages

These ApID ranges are specified in the Swift 1553 Bus Protocol Interface Control Document.
Note: RT-RT commands are addressed in section 4.12.4

4.7.4 Formatting and Output of Large Science Data Products

-1 The flight software shall accept large data files from the science software to be sent to the spacecraft data recorder.

-2 The flight software shall break the large science data files into packets according to the format specified in the BAT Flight Software to Science Data Center ICD.

-3 Since the BAT can generate data much faster than it can be sent to the spacecraft, the flight software shall maintain a queue of science data files that are ready to send to the spacecraft.

-4 The output queue shall be large enough to hold all of the data associated with a single gamma ray burst

-5 Each large science data file shall have a priority. The flight software shall always choose the highest priority file in the queue to output.

-6 Once the software has started to output a file, it shall finish sending that file before starting another file, even if a newly available file has higher priority than the current file being sent.

-7 Science file output operations shall not interrupt the output of normal housekeeping telemetry.

4.8 FILE SYSTEM MANAGEMENT

The following sections detail requirements related to the management of the onboard file systems. This includes transfers of files between on-board disks as well as between the spacecraft and ground.

4.8.1 File Uplink

-1 The flight software shall accept files uplinked from the ground to be stored on any of the on-board file systems.

-2 The flight software shall accept uplinked files as a series of command packets containing gnu "cpio" formatted data.

-3 The flight software shall report the status of file uplinks in housekeeping data, including at least the following:
filename of the current/last uploaded file
total size of the current/last uploaded file
current bytes received for the current uploading file

4.8.2 File Downlink

-1 The flight software shall be able to downlink files to the ground from any of the on-board file systems on command.

-2 Downlink file data shall be in "cpio" format contained in CCSDS packets

-3 There shall be two Application IDs used for file downlink. One ApID will be for real-time file downlink, and one for files sent to the data recorder.

-4 The flight software shall maintain a queue of files ready to be downlinked

-5 Each file in the downlink queue shall have a priority. The flight software shall always downlink the highest priority file in the queue

-6 New files placed in the downlink queue shall not interrupt the currently downlinking file, even if they have higher priority. New file downlink shall start only after the current file is finished.

-7 File downlink operations shall not interrupt the output of normal housekeeping telemetry.

4.8.3 On-board File Transfers

-1 The flight software shall provide ground commands to copy, move, and delete files on any of the on-board file systems

-2 The flight software shall provide ground commands to create, remove, and list the contents of directories on any of the on-board file systems.

-3 The flight software shall report the status of each on-board file system in housekeeping telemetry. This status shall include at least the amount of free space for each file system.

4.9 BAT TIME MANAGEMENT

The following sections detail requirements related to the onboard time. This includes time maintenance and distribution.

4.9.1 Instrument Time Maintenance and Distribution

-1 The flight software shall maintain instrument time and be capable of distributing it to onboard subsystems when requested. The time code format shall comply with the CCSDS Unsegmented Time Code Specifications.

-2 Current GMT time shall be derived from the 1Hz interrupt and 1Hz time message from the spacecraft.

-3 The flight software shall forward the spacecraft 1Hz time message to the BCDHs within 50ms of receipt.

-4 If the spacecraft stops sending the 1Hz time message, the BAT flight software shall generate and forward its own best estimate of the time to the BCDHs once per second.

-5 The flight software shall use one of the programmable interval timers on the RAD6000 processor card to determine time to sub-second precision.

-6 The flight software shall time stamp telemetry packets with an accuracy of +/-100ms with respect to the GMT provided by the spacecraft.

-7 The maintenance of instrument time shall not be adversely affected by cold or warm restarts of the flight software.

-8 Any status related to time maintenance shall be available in housekeeping data.

4.10 BAT INSTRUMENT SAFING

This section addresses those requirements related to BAT health and safety. Subsections detail the requirements for the flight software's management of the instrument's watchdog strategy, the support for onboard diagnostics, the generation of housekeeping telemetry, the monitoring of telemetry and memory data points, the handling and responses to single event upsets and the management of bulk memory data storage in response to certain safing conditions.

4.10.1 Watchdog Management

-1 The flight software shall service the processor watchdog timer once every second, setting it to expire in 10 seconds.

4.10.2 Onboard Diagnostics

-1 The flight software shall support self-test of memory and hardware components.

-2 The flight software shall not perform diagnostics during initialization. The flight software shall support the execution of self tests by ground command only.

-3 The flight software shall provide the results of any self test diagnostics in housekeeping telemetry data. The flight software shall provide the status of any executing self test ground commands in housekeeping telemetry data.

-4 The flight software shall execute self tests such that processor time is not taken away from critical instrument operations.

-5 The flight software shall be capable of running self test operations on the following devices and/or functional interfaces:

Local RAD6000 RAM
Local RAD6000 EEPORM
Bulk Memory
External EEPROM (on bulk memory card)
1553 Shared RAM
Multi-channel I/F Card RAM
EDAC Functions
DMA Functions
1553 Protocol Chip

4.10.3 <u>Housekeeping Telemetry</u>

-1 The flight software shall maintain housekeeping status for each software subsystem. Housekeeping status shall include processing status, configuration parameters, critical processing variables and error indicators as a minimum.

-2 The flight software shall not perform any subcomming of data within housekeeping. Each housekeeping data field shall have the same meaning for each packet.

-3 Each flight software subsystem shall be capable of resetting parts of its housekeeping data by ground command.

-4 The flight software shall be able to combine housekeeping packets from various subsystems into larger packets. The arrangement of data from the smaller packets in the larger packet shall be table driven.

-5 The flight software shall include flags in housekeeping to indicate whether or not the data for a particular subsystem has been updated since the last housekeeping packet was generated.

-6 If a task fails to respond to two housekeeping requests in a row, the flight software shall be able to take any of the following actions:

issue an event message
perform a reset of an individual task
perform a reset of the DSP board
perform a reset of the entire system

The appropriate response for each task shall be configurable through a table load.

-7 The flight software shall tag the resulting combined housekeeping packet with the time it was built.

-8 As defined in the appropriate Interface Control Documents, the flight software shall read and output housekeeping data from external hardware devices.

4.10.4 Health and Safety Monitoring

The requirements listed in this section relate to the monitoring of BAT health and safety. The functions addressed below are performed autonomously by the flight software to ensure the health and safety of the BAT during normal activities. Sections discussed include the monitoring of telemetry and memory data points as well as memory dwell, checksum and scrub capabilities.

4.10.4.1 Telemetry and Memory Monitoring

-1 The flight software shall provide the capability to perform general-purpose health and safety monitoring functions. This monitor shall be able to process any telemetry point, memory address or I/O port and initiate a configured response upon detection of a constraint violation.

*Note: The following requirements use the generic term **watchpoint** to refer to a single telemetry value, memory address or I/O port being monitored by the flight software.*

-2 The flight software shall perform watchpoint monitoring and evaluation during all modes. Telemetry watchpoint evaluation shall be performed whenever a new packet containing the data is received. Memory and I/O watchpoints shall be evaluated at 1 second intervals.

-3 The flight software shall be capable of monitoring at least 128 different watchpoints. The definition of each watchpoint shall include the following items:
location and type of the watchpoint
bit size of the watchpoint (1-8, 16, 32, or 64)
watchpoint bitmask value

-4 The flight software shall be able to perform logical comparison operations of current watchpoint values with pre-defined constant values. At a minimum, the software must support "less than", "greater than", and "equal to" operations.

-5 The flight software shall allow for logical combinations of up to 4 watchpoints to define the conditions for a constraint violation.

-6 The flight software shall be able to take autonomous corrective actions when constraint violations are detected.

-7 The response to constraint violations shall be configurable and shall include the ability to initiate any instrument command or series of commands.

-8 The flight software shall be capable of responding to monitored constraint violations within 1 second of receipt of the telemetry packet containing the monitored value.

-9 The flight software shall provide a means to disable all constraint checking, as well as individual constraints or groups of related constraints.

-10 The flight software shall be able to issue an event message when a constraint violation occurs.

-11 The flight software shall report the status of constraint checking in housekeeping telemetry.

4.10.4.2 Instrument State of Health Monitoring

In addition to the general monitoring capabilities in the previous section, the flight software will require some BAT-specific state of health monitors. These are included here mainly because they are too numerous to be included in the generic monitoring scheme described above.

-1 The flight software shall monitor the following temperatures and voltages in the housekeeping data from the power interface box and each BCDH:

    BCDH current (from power box)
    BCDH power converter temperature (from power box)
    BCDH power converter +5V output (from power box)
    BCDH power converter +3.3V output (from power box)
    BCDH power converter +12V output (from power box)
    BCDH power converter -12V output (from power box)
    Block mechanical temperature (from power box)
    Block front end temperature (from power box)
    Block heater controller setpoint (from power box)
    Block electronics temperatures, 4 per block (from BCDH)
    Block leakage currents, 8 per block (from BCDH)
    +2.5 Voltage regulator input (from BCDH)
    -2.5 Voltage regulator input (from BCDH)

-2 The flight software shall contain a configurable upper and lower limit for each temperature or voltage value. These limits will be the same value for all BCDHs, but may be different for each different temperature or voltage.

-3 The flight software shall check each of the temperatures and voltages against their limits every N seconds, where N is a configurable parameter.

-4 The flight software shall be able to perform a configurable response when a temperature or voltage is out of range. The response may be any combination of the following actions:

    send an event message reporting the error condition and BCDH number
    send a TDRSS message reporting the error condition and BCDH number
    reset the affected BCDH
    power down the affected BCDH
    set the HV Control DAC to zero for the affected DM

-5 The configurable response will be the same for all BCDHs, but may be different for each different temperature or voltage. The response shall be targeted to only the BCDH whose value is out of range.

-6 The limit values and associated responses shall be defined in a flight software system table.

-7 The flight software shall provide commands to disable/enable monitoring for:

    a single value in a single BCDH,
    all values in a single BCDH,
    a single value for all BCDHs, or
    all values for all BCDHs.

4.10.4.3 Memory Dwell

-1 The flight software shall be capable of periodically monitoring addressable memory locations. This functionality shall be available in all operating modes.

-2 The flight software shall be capable of enabling or disabling the memory dwell functions by ground command.

-3 The flight software shall allow up to 64 memory addresses to be configured for the memory dwell functions and shall provide a limited capability for controlling the sample rate for each configured address. This configuration data shall be stored in a system table to facilitate easy modification by ground command.

-4 The flight software shall allow sampling of configured memory addresses to be as frequent as every 100ms.

-5 The flight software shall limit, as required, the number of memory addresses that can be sampled in one second to conserve overall CPU utilization.

-6 The flight software shall telemeter the status of memory dwell functions to the ground.

4.10.4.4 Memory Checksum

-1 The flight software shall perform checksum validation on addressable regions of memory whose contents are known to not change over time such as EEPROM space and RAM locations containing static data and/or executable code.

-2 The flight software shall reference the master system table to determine which regions of memory to checksum. This table shall include an expected checksum value for each memory region.

-3 The flight software shall compute the memory checksum by accumulating the sum of 8 bit quantities while storing the result in a 16 bit unsigned value. The initial checksum value shall be zero.

-4 The flight software shall be capable of enabling or disabling the memory checksum process by ground command.

-5 If memory checksumming is enabled and the calculated checksum differs from the master checksum found in the master system table, the flight software shall generate a checksum error event message.

-6 If a checksum error is detected in a region of memory marked as *recoverable on-orbit*, the flight software shall re-copy that area from EEPROM to RAM in an attempt to clear the error.

-7 Once checksumming is enabled, the appropriate software subsystems shall coordinate such that the loading of new data into memory regions being checksummed does not result in erroneous checksum errors being reported.

-8 The flight software shall be capable of completing a full cycle of checksum validation on the configured memory regions in 15 minutes.

-9 Checksum errors shall be reported to the ground in housekeeping telemetry and as significant event messages.

4.10.4.5 Bulk Memory Scrubbing

-1 The flight software shall be capable of scrubbing all addressable bulk memory to correct single bit errors and to detect multibit errors. This process shall work in conjunction with the hardware's EDAC feature. *(Scrubbing is the act of reading then writing memory locations.)*

-2 The flight software shall provide the capability to enable and disable the EDAC functionality of the memory hardware.

-3 The flight software shall report any single bit errors to the ground in housekeeping data.

-4 The flight software shall report any multibit errors to the ground in telemetry as system event messages. The flight software shall ensure that a *flood* of multibit errors does not adversely affect instrument operations.

-5 After a restart, the flight software shall determine whether power has been cycled on the bulk memory, thus corrupting the EDAC checkbits. If the flight software determines that the checkbits are corrupt, it shall re-initialize the checkbits associated with all bulk memory partitions.

-6 The flight software shall provide memory scrub status in housekeeping data. This status shall include, as a minimum, the following items:

  number of single bit errors
  number of uncorrectable multibit errors
  address of last multibit error
  number of complete scrub passes

-7 The flight software shall be capable of completely scrubbing bulk memory at least once every 90 minutes (~1 orbit).

4.10.5 <u>SEU and Error Handling and Reporting</u>

It is a goal that the BAT FSW be capable of safely recovering from any Single Event Upset (SEU) which affects instrument functions. However, since SEUs are inherently unpredictable, it is impossible to prove that the FSW meets this goal. This document does attempt to address most types of SEUs. SEUs that cause corruption of either code or static data areas of RAM are handled by periodic checksumming (section 4.10.4.4). SEUs in bulk memory are handled by memory scrubbing (section 4.10.4.5). SEUs that cause a single task to "lock up" are handled through the periodic housekeeping collection process (section 4.10.3). SEUs that cause the entire system to "lock up" are handled using the hardware watchdog timer (section 4.10.1). This section covers detectable SEUs other than the four types mentioned above. These SEUs are handled using "software event" messages. Conditions causing software event messages include (but are not limited to):

  system resets
  rejected commands (bad command parameters or state)
  unexpected hardware behavior
  non-recoverable EDAC errors (multi-bit)

-1 When the flight software detects an error or significant event, it shall generate a software event message.

-2 Each software event message shall contain a code number that identifies the event and some additional data that may help to debug the problem.

-3 In response to each software event message, the flight software shall be capable of performing a warm or cold restart.

-4 The flight software response to each software event shall be configurable by table upload

-5 Each software event message shall be routed for downlink and bulk memory storage. The time tag in each significant event message shall indicate the time at which the event occurred or was first detected by the flight software.

-6 The flight software shall manage event generation so that numerous, repetitive significant event messages do not flood the telemetry link or adversely affect instrument operations.

4.11 ONBOARD SOFTWARE MAINTENANCE

The requirements in this section deal with the topic of onboard software maintenance. Onboard software maintenance, at the lowest level, pertains to the capabilities for reading and writing to various regions of memory available to the flight software. The section on software loading lists the requirements related to reprogramming the boot and normal mode software on the instrument. The memory management section lists generic requirements related to reading and writing memory locations and the last section discusses requirements for loading and dumping system tables. A system table is an area of memory that has been conveniently defined to allow easy reading and writing of its contents.

4.11.1 Software Load Capabilities

-1 During development and test activities, it shall be possible to load new flight software into EEPROM over the processor's serial interface.

-2 While on-orbit, it shall be possible to load new enhanced mode flight software into EEPROM using ground commands.

-3 While on-orbit, it shall not be possible to write new flight software to the launch mode EEPROMs.

4.11.2 Memory Management

-1 The FSW shall be capable of reading from all memory or I/O space in response to memory dump ground commands.

-2 The parameters within memory dump ground commands shall be validated before performing the requested action. Validation errors shall be reported to the ground.

-3 Large memory dumps shall be formatted as ITOS-style dump image files. These files may be created anywhere on the on-board file system, or dumped directly to the ground.

-4 For small (up to 200 bytes) memory dumps, the data shall be formatted as a fixed length CCSDS packet. For dumps less than 200 bytes, the rest of the packet shall be filled with 0s.

-5 The format of the actual dump data in a memory dump packet shall conform to the standard dump packet format defined in the Swift T&C Handbook.

-6 The status of memory dump operations shall be available in housekeeping.

-7 The flight software shall be capable of writing to memory or I/O space in response to memory load ground commands.

-8 The flight software shall be able to load the following types of memory:
local processor RAM
bulk RAM

1553 shared RAM
MIC card RAM
write-able EEPROM

The memory load command will specify the type of memory being addressed.

-9 The parameters within memory load ground commands shall be validated before performing the requested action. Validation errors shall be reported to the ground.

-10 If a memory load command affects a static area of memory processed as part of the flight software's memory checksumming activities, the flight software shall not automatically recompute the checksum of the affected area of memory. (It is the responsibility of the ground operators to uplink a new checksum for that memory area.)

-11 The FSW shall provide a method for reading and writing a single byte, 16bit word, or 32bit word.

4.11.3 Table Management

-1 The flight software shall support the capability to format configuration parameters into system tables, which can be stored in onboard memory.

-2 The default startup values for each system table shall be stored in EEPROM. Each system table shall be managed by a particular software subsystem. During a cold start, each software subsystem shall copy its system tables from EEPROM to RAM.

-3 The configuration information for all system tables shall be stored in a master table, which shall itself be formatted as a system table.

-4 The master table shall contain a field for each system table indicating if the table is write locked. It shall not be possible to load a new version of a table if it is write locked.

-5 The master table shall contain a field for the checksum of each system table.

-6 The flight software shall be capable of reading any system table in response to table dump ground commands.

-7 The parameters within table dump ground commands shall be validated before performing the requested action. Validation errors shall be reported to the ground.

-8 Table dump ground commands shall only have to specify a single token value to identify which system table the command is requesting be dumped.

-9 It shall be possible to dump the complete contents of a system table.

-10 It shall be possible to dump individual pieces of a system table. Partial table dump ground commands shall specify the element within the table from which to dump.

-11 The status of a currently active (or last completed) table dump shall be available in housekeeping.

-12 Table dumps shall be formatted as ITOS-style dump image files. These files may be created anywhere on the on-board file system, or dumped directly to the ground.

-13 The flight software shall be capable of storing new versions of system tables in response to table load ground commands.

-14 The parameters within table load ground commands shall be validated before performing the requested action. Validation errors shall be reported to the ground.

-15 Table load ground commands shall only have to specify a single token value to identify which system table the command is requested to load.

-16 It shall be possible to load the complete contents of a system table.

-17 It shall be possible to load individual pieces of a system table. Partial table load ground commands shall specify the element within the table where the new data is to be loaded.

-18 The flight software shall support the capability to load a new version of a system table into its currently active RAM location or its default EEPROM location.

-19 The flight software shall reject table load commands if the table being loaded is marked as write locked in the master table.

-20 The flight software shall provide a method to allow the new table data to be validated by the appropriate software subsystem to ensure the new table is acceptable. The flight software shall only copy, or commit, the new table data to its final destination if the validation is successful.

-21 In response to each table load activity, the flight software shall compute and output the checksum of the full table. This checksum shall become the new master checksum used by the onboard memory checksum software.

-22 If the table load command contains a checksum, the flight software shall validate it before actually committing the load data to its final destination.

-23 The ground shall be notified when a table load and its associated activities are done.

-24 Flight software housekeeping shall contain data indicating the status of a currently active table load or last completed table load.

4.12 BAT FSW 1553 REMOTE TERMINAL INTERFACE SOFTWARE REQUIREMENTS

4.12.1 General

-1 The RT software shall configure as a 1553 Remote Terminal in accordance with the MIL-STD-1553B protocol.

-2 BAT shall be configured as a Remote Terminal (RT) under all modes of operation. (1553 ICD 3.1)

-3 The RT software shall initialize and control the UTMC Summit device and the 1553 shared memory.

-4 The RT software shall support bi-directional communications between the BAT software subsystems and the spacecraft via the 1553 bus.

-5 The RT software shall interface with the BAT software subsystems via the C&DH software bus.

-6 The RT software will support data wraparound as defined in the SWIFT 1553 BUS PROTOCOL ICD section 3.1.4.

4.12.2 Commands

-1 The RT software shall receive telecommands from the 1553 remote terminal as defined in the SWIFT 1553 BUS PROTOCOL ICD section 4.3.

-2 The RT software shall truncate commands to the length specified in the CCSDS packet header before forwarding to the command destination via the software bus.

-3 The RT software shall maintain a telemetry counter to indicate the number of command packets received from the spacecraft.

4.12.3 Telemetry

-1 The RT software shall transfer science and housekeeping telemetry packets to the spacecraft via the 1553 bus as defined in the SWIFT 1553 BUS PROTOCOL ICD section 4.8.

-2 The RT software shall update the 1553 telemetry data buffers and subsequently increment the transfer request counter whenever a word is received from the BC on the "done" subaddress # 26. The BAT RT software shall timeout whenever the BAT RT receives no "done" from the spacecraft for a period of TBS seconds. After a timeout, the RT software shall update the telemetry buffers with new telemetry data and subsequently increment the transfer request counter.

-3 The RT software shall maintain a telemetry counter to indicate the number of telemetry packets transferred to the spacecraft.

4.12.4 RT/RT Transfers

-1 The RT software shall support RT/RT transfers as defined in the SWIFT 1553 BUS PROTOCOL ICD section 4.4.

-2 The RT software will provide at least 2 levels of RT/RT transfer priorities. Messages of higher priority will be transferred before messages of lower priority regardless of the amount of time the lower priority message has been pending. Note - this will expedite transfer of hi-priority commands (e.g. FOM commands).

-3 The RT software shall maintain a telemetry counter to indicate the number of RT/RT command packets transferred to the spacecraft.

4.12.5 RT Task commands

-1 The RT software shall respond to transactions on either 1553 bus by default. The remote terminal software shall support ground commands to disable responses on either bus, but will not allow both busses to be disabled at the same time.

-2 The RT software shall support a command to reset all the housekeeping counters to zero.

4.12.6 RT Task Telemetry

-1 The RT software shall report 1553 processing and control status to the ground via housekeeping data. As a minimum, the following items shall be reported:

Which bus(es) are active
Number of packets received from the bus controller
Number of telemetry packets sent to the bus controller
Number of RT/RT commands sent to the bus controller

4.13 Digital Signal Processor Core Software

The Digital Signal Processor (DSP) is responsible for running various science processing routines, as specified in section 3. This section describes the requirements for the engineering support software necessary to get input data from the RAD6000, start the science routines, and send the results back to the RAD6000.

-1 After power up, the DSP core software shall initialize the hardware and begin looking for requests from the RAD600 within 30 seconds.

-2 The DSP core software shall poll for requests from the RAD6000 by monitoring a specific address in shared memory for an incrementing command counter.

-3 Ground commands to the DSP shall follow the same CCSDS format as commands to the RAD6000.

-4 When not currently running a science routine, the DSP shall poll for a new request from the RAD6000 at least once every 10ms.

-5 The DSP core software shall provide a special poll function to allow higher priority science routines to pre-empt lower priority routines. Lower priority science routines must call this poll function periodically as they run, and exit immediately if the poll function returns "true".

-6 Each request from the RAD6000 shall contain a "function code" defining the type of request. Core software commands will have function codes 0-63, low priority (preemptable) science function codes will be 64-127, and high priority (uninterruptible) science function codes will be 128-255.

-7 Each request from the RAD6000 may also contain a data pointer and a data size. The RAD6000 software shall copy this data from the RAD6000 memory to DSP local memory before sending the command to the DSP.

-8 For function codes above 63, the core software shall call a single science processing routine, passing it the function code, data size, and pointer to the data in local DSP memory.

-9 When a DSP function is complete, the core software shall place the response data in a fixed location in shared memory, and then interrupt the RAD6000.

-10 The DSP core software shall maintain a buffer of DSP housekeeping data in a fixed location in shared RAM. The RAD6000 can read this HK data at any time.

-11 Response data sent from the DSP to the RAD6000 shall be in the form of CCSDS packets.

-12 The DSP shall be able to generate system event packets to report significant events.

-13 For software maintenance purposes, the DSP core software shall provide the ability to modify code areas in flight. The DSP software shall handle the address translation to the 48bit code space.

-14 When not otherwise occupied, the DSP shall perform checksumming on code and static data areas in RAM, looking for SEUs. If an SEU is detected the DSP shall inform the RAD6000 through an event message.

4.14 POWER MANAGEMENT

This section describes the software requirements relating to the interface with the BAT power box. More details on the interface are found in the BAT IP to BAT Power Box ICD.

4.14.1 Power Box Commands

-1 Power box commands will be CCSDS command packets containing a packet header, function code and 16 bits of command data. All power box commands will have the same Application ID.

-2 The flight software shall strip off the CCSDS packet header and send the function code (1 byte) and data field (2 bytes) over the serial link to the power box. The flight software shall follow each three byte command with a one byte checksum, calculated according to the *BAT Power Box to Instrument Processor ICD*.

-3 After receiving each command, the power box will respond with a status message containing a command echo. The flight software shall validate that the command echo matches the last command sent, and issue an event message if the validation fails.

-4 By default, the flight software shall command the power box on the A side of the serial I/F. If the power box fails to respond to a command within 1 second, the flight software shall issue an event message and autonomously switch to the B side of the serial I/F. The flight software shall also provide a command to manually switch the active side of the serial I/F.

-5 The flight software shall autonomously send the power box command to request standard housekeeping data on a periodic basis (nominally every 5 seconds).

-6 On command, the flight software will begin continuously requesting diagnostic dwell packets from the power box. This diagnostic command will contain the following fields:
The MUX channel for the dwell
The total number of packets to request
An optional PCB command to be executed during the collection of data for the first dwell housekeeping packet.

-7 Normal power box housekeeping collection will cease while power box diagnostics are in progress.

-8 On receiving a "cancel" command, the flight software will stop requesting diagnostic packets and return to the standard housekeeping request cycle.

4.14.2 Power Box Telemetry

-1 The flight software shall read data from the power box serial link and format it into CCSDS packets.

-2 Telemetry responses from the power box will come in one of two fixed lengths (short form or long form). For short form responses the flight software shall discard the data after validating the command as specified in 4.14.1-3. For long form responses the flight software shall add the CCSDS headers, fill in the timestamp, and output the resulting packet.

-3 The flight software shall allow for two different Application IDs on power box packets, one for standard housekeeping data, and one for diagnostic data.

-4 The flight software shall maintain information about the last command sent to the power box, so it can determine which ApID to use on the power box telemetry packet. (The power box will not send data on the serial link autonomously, so the flight software will always know what was last requested.)

-5 The flight software shall report status of the power box I/F in housekeeping telemetry, including at a minimum:

The current side of the I/F in use (A or B)
The total number of standard housekeeping data packets requested
The total number of power box commands sent (excluding HK data requests)
All four bytes of the last power box command sent (excluding HK data requests)
The total number of command echoes that failed validation

4.15 BCDH SOFTWARE INTERFACE

The term “BCDH Software Interface” is shorthand for the interface between the BAT C&DH Software and the detector modules (DM), the Block Command & Data Handlers (BCDH), and the Multi-channel Interface Card (MIC). The interface passes commands to the hardware and controls the autonomous delivery of data by the hardware into bulk memory. Thus, the interface provides a method for software to meet the following requirements:

4.15.1 Hardware Calibration

Refer to section “BAT Science Code Requirements” sub-section “Calibration Requirements” for a detailed description of software requirements related to hardware calibration.

4.15.2 Interface Commands

The BCDH Software Interface shall respond to the following interface commands:

-1 No-operation (NOP) command. Performs no function other than to increment the successful commands counter maintained in the interface housekeeping telemetry.

-2 Reset housekeeping counters command. This command shall cause software to clear to zero all interface housekeeping telemetry status counters.

-3 Hardware initialization command. Software shall initialize the MIC/BCDH/DM hardware to the values contained in the hardware initialization table. See section 1.1.1.3 “Hardware Initialization” for a description of the hardware initialization table. The scope of the initialization shall be programmable and shall include (but not be limited to) the following options:

The entire hardware array
The MIC card configuration registers
One, or more, of the 16 BCDH’s
One, or more, of the 8 DM’s within each of one, or more, BCDH’s

-4 Hardware status command. Software shall provide a method to command the MIC/BCDH/DM hardware to generate status telemetry packets. See section 1.1.1.8 “Hardware Status Telemetry” for a description of the hardware status telemetry data points. The scope of the status collection shall be programmable and shall include (but not be limited to) the following options:

The entire hardware array
The MIC card

One, or more, of the 16 BCDH’s
One, or more, of the 8 DM’s within each of one, or more, BCDH’s

-5 Pass through commands. Software shall provide a method to deliver command packets to the MIC/BCDH/DM hardware. (note: the contents of pass through command packets are not examined by software)

-6 Software mapped hardware commands. Software shall provide entry points to functions that provide all commonly used hardware functions including:
Reset detector module
Enable/disable individual detectors
Set detector module(s) into calibration mode
Set voltage levels

-7 Monitor hardware state changes. Software shall monitor the software mapped hardware commands and modify the hardware initialization table to reflect changes to the current state for the following elements:
(TBD – list is subset of hardware initialization table)

-8 Write MIC configuration registers. Software shall provide a method to write to the following MIC configuration registers:
Link configuration registers (2 per each of 16 BCDH’s)
Bulk memory circular buffer start address
Bulk memory circular buffer length

4.15.3 <u>Hardware Initialization</u>

The BCDH Software Interface shall provide a method to initialize the MIC/BCDH/DM hardware. The specific hardware initialization requirements are:

-1 Software shall initialize the following MIC configuration settings:
Link configuration registers (two per each of 16 BCDH’s)
Link control enable switch (one per each of 16 BCDH’s)
Frequency of DMA data transfers from link FIFO’s to bulk memory

-2 Software shall initialize the following MIC configuration settings:
Bulk memory circular buffer start address
Bulk memory circular buffer length

-3 For each of the 16 BCDH’s, software shall initialize the following:
FIFO programmable flag offset
FIFO read/write threshold
Autonomous timestamp packet enable switch
Autonomous event count packet enable switch
SpaceWire link transmit rate

-4 For each of the 16 BCDH’s, software shall initialize the following:
DAC common bias voltage (pre-amplifier)
DAC common bias voltage (shaping amplifier)
DAC common bias voltage (general bias)
DAC common bias voltage (DAC bias)
DAC common bias voltage (trigger delay)
DAC common bias voltage (reset width)
DAC common bias voltage (ADC latchup threshold)
DAC common bias voltage (XA1 latchup threshold)

-5 For each of the 16 BCDH's, software shall initialize the following:

DAC high voltage control (DM number 0)
DAC high voltage control (DM number 1)
DAC high voltage control (DM number 2)
DAC high voltage control (DM number 3)
DAC high voltage control (DM number 4)
DAC high voltage control (DM number 5)
DAC high voltage control (DM number 6)
DAC high voltage control (DM number 7)
High voltage power enable switch (must follow DM initialization)

-6 For each of the 128 DM's, software shall initialize the following:

DAC voltage setting (calibration pulser)
DAC voltage setting (feedback shaper)
Tagged source coincidence checking delay period
XA1 power enable switch

-7 For each of the 128 DM's, software shall initialize the following items at least 10 milliseconds after the BCDHs were initialized:

XA1 automatic latchup shutdown enable switch
XA1 automatic latchup alarm packets enable switch

-8 For each of the 256 DM "sides", software shall initialize the following:

Event filter mask
Operation mode
Calibration pulser period
XA1 version number register
Channel control register – 128 channel enable switches

-9 For each of the 256 DM "sides", software shall initialize the following:

Segment gain coefficient
Segment offset coefficient
Strip gain coefficient
Strip offset coefficient

-10 For each of the 256 DM "sides", software shall initialize the following:

DAC voltage setting (feedback pre-amp)
DAC voltage setting (discriminator threshold)
DAC voltage setting (output buffer offset adjust)

-11 For each of the 256 DM "sides", software shall initialize the following items at least 10 milliseconds after the BCDHs were initialized:

ADC automatic latchup reset enable switch
ADC automatic latchup alarm packets enable switch

-12 Software shall provide conditionally compiled code to perform the following initialization for the ETU boards only:

For each of the 8 ETU DM's, configure Xilinx FPGA through BCDH

-13 Software shall maintain the list of MIC/BCDH/DM hardware initialization values as a table. Refer to section 4.11.3 "Table Management" of this document for a description of common C&DH software table attributes.

4.15.4 Detector Module Latchup Mitigation

Refer to the document "BAT Detector Module Controller Specification" for a detailed description of the latchup detection and mitigation strategies regarding the XA1 registers and ADC circuits. The specific latchup mitigation requirements are:

-1 XA1 Latchup Mitigation. Software shall provide the capability to restore the affected DM to the state maintained in the hardware initialization table.

-2 ADC Latchup Mitigation. Software shall provide the capability to restore the affected DM to the state maintained in the hardware initialization table.

4.15.5 Periodic DAC Refresh

Detector array DAC voltage control registers require periodic refresh from internal shadow registers for SEU mitigation. Software shall invoke continuous refresh of the DAC voltage control registers. The specific DAC refresh requirements are:

-1 Single command. The DAC refresh command shall make use of the command routing facilities of the MIC and BCDH's to send a single command that is subsequently routed to each of the currently enabled detector modules.

-2 Continuous loop. The DAC refresh command shall repeat every 5 seconds by default.

-3 Programmable loop rate. The frequency at which the DAC refresh command is generated shall be modifiable from the ground in the range from once per second to once per orbit (90 min).

4.15.6 Time Distribution

Refer to section 4.9 "BAT Time Management" for a complete description of the BAT instrument time management requirements. For convenience, the following time related software requirement is also displayed at this location:

-1 Absolute timestamp. Software shall forward to each of the 16 BCDH's the 1 Hz timestamp packet received from the spacecraft C&DH.

4.15.7 Interface Status Telemetry

The BCDH Software Interface shall provide the following status telemetry data:

-1 Interface statistics: Software shall maintain a record of operational statistics including command and error counters in addition to miscellaneous data sufficient to describe the general health and status of the BCDH Software Interface.

-2 Housekeeping telemetry packet: Software shall respond to requests from the Health and Safety task by generating a housekeeping telemetry packet.

-3 Hardware status registers. Software shall collect hardware status data from the MIC card. (note: this data is exclusive of status packets which the MIC can be commanded to deliver into the science data stream) The data collected shall include, but not be limited to, the following:
  Link configuration registers (2 per each of 16 BCDH's)
  Bulk memory circular buffer start address

Bulk memory circular buffer current address
Bulk memory circular buffer length

4.15.8 Hardware Status Telemetry

Software shall provide a method to command the MIC/BCDH/DM hardware to report status telemetry. Note that the frequency for collecting each particular data point is an operational issue that will be determined during development. The specific data point requirements are:

-1 Software shall provide a method to command the MIC to generate the following status data:
MIC status register
16 Link status registers

-2 Software shall provide a method to command each of the 16 BCDH's to generate the following status data:
BCDH status packet
Common bias voltages
High voltage levels
DM power levels
BCDH power levels
Board temperatures
Voltage reference levels
High voltage reference levels

-3 Software shall provide a method to command each of the 128 DM's to generate the following status data:
Housekeeping packet
Channel control register
DAC shadow register
XA1 shadow register

-4 Software shall provide a method to command each of the 256 DM sides to generate the following status data:
Status packet
Status counters packet
Segment gain coefficient
Segment offset coefficient
Strip gain coefficient
Strip offset coefficient
Parameters packet
CCR data
DMR data
XSR data

5 RESOURCE REQUIREMENTS

This section contains the resource requirements related to the flight software and its associated hardware components. The subsections discuss memory usage, timing requirements and utilization of the RAD6000 processor, and the PCI backplane.

5.1 MEMORY USAGE

-1 The launch mode EEPROMs shall have a minimum of 10% spare capacity.

-2 The enhanced mode EEPROMs shall have a minimum of 20% spare capacity.

-3 local processor RAM utilization by the flight software shall not exceed 90%.

5.2 TIMING REQUIREMENTS

-1 To preserve the time tagging accuracy of data, the flight software shall begin the scheduling of system activity within 500usec of resynchronizing with the spacecraft 1Hz clock following a processor restart.

-2 The accuracy of the time tag in telemetry packets shall be +/- 100ms, relative to the spacecraft clock.

5.3 PROCESSOR UTILIZATION

-1 The processor shall never exceed 80% utilization, measured over a 5 second interval.

5.4 PCI BACKPLANE UTILIZATION

-1 Utilization of the PCI backplane shall not exceed 80%, measured over a 5 second interval.

6 Appendix A - SYSTEM REQUIREMENTS TRACEABLITY MATRIX